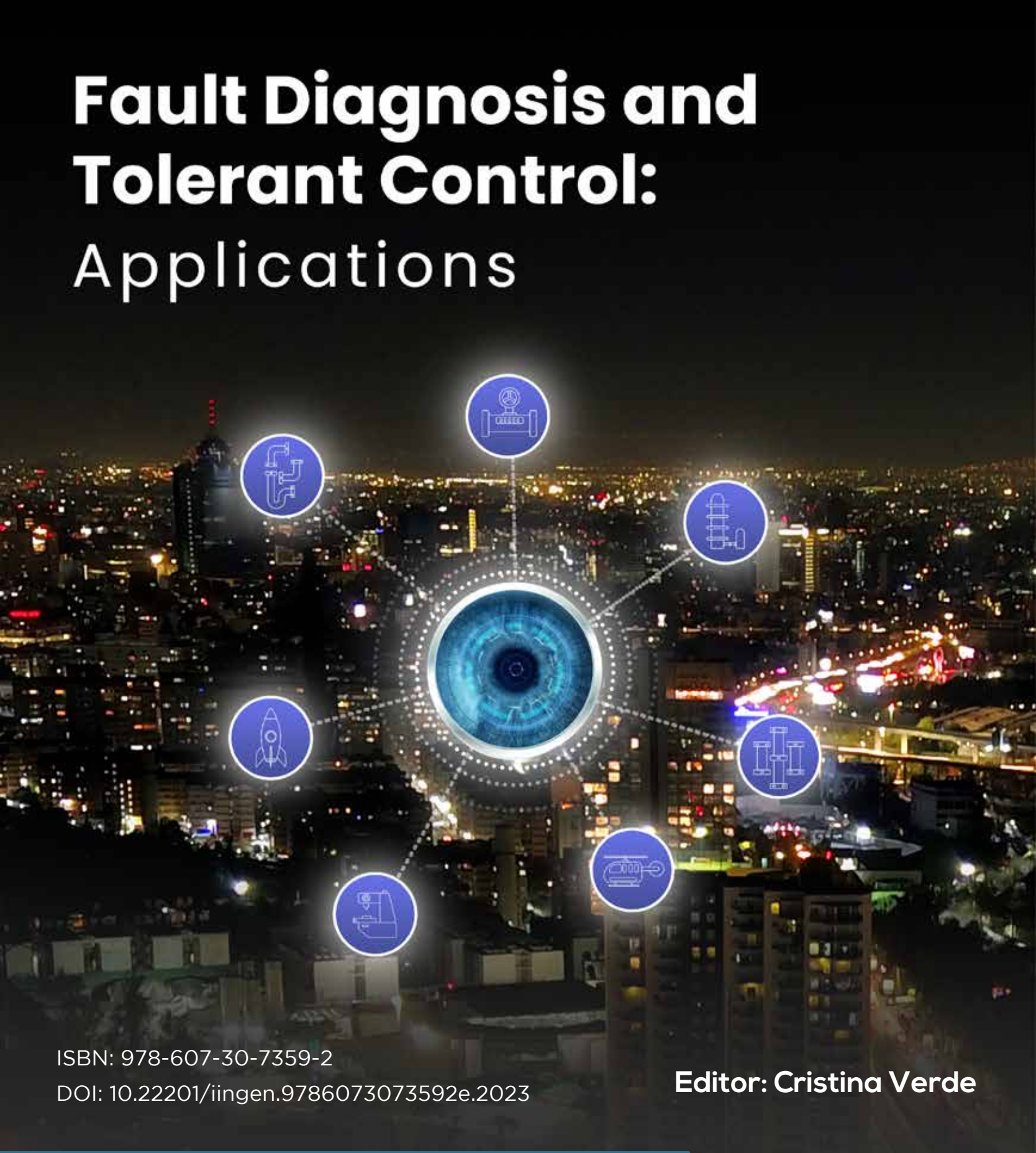
Fault Diagnosis and Tolerant Control: Applications



ISBN: 978-607-30-7359-2
DOI: 10.22201/iingen.9786073073592e.2023

Editor: Cristina Verde

INSTITUTO DE INGENIERÍA UNAM

Serie
Investigación y Desarrollo
Publicación arbitrada

SID 713
Marzo, 2023

This page intentionally left blank

Publicación arbitrada
ISBN: 978-607-30-7359-2
doi: https://doi.org/10.22201/iingen.9786073073592e.2023

Fault Diagnosis and Tolerant Control: Applications

Cristina Verde
Editor[1]

Serie de Investigación y Desarrollo
SID 713
Marzo 2023

[1] Investigador Titular C, Instituto de Ingeniería UNAM
verde@unam.mx

Fault Diagnosis and Tolerant Control: Applications

Primera edición, marzo de 2023
SID 713

Instituto de Ingeniería, UNAM
Ciudad Universitaria, CP 04510, Ciudad de México

ISBN: 978-607-30-7359-2
doi: https://doi.org/10.22201/iingen.9786073073592e.2023
La obra fue editada por el Instituto de Ingeniería, de la Universidad Autónoma de México (IIUNAM). El cuidado de la edición estuvo a cargo de la Unidad de Promoción y Comunicación del IIUNAM. Esta obra está gratuitamente disponible para consulta e impresión, en archivo PDF de 13.6 MB, en la sección de Publicaciones del portal electrónico del IIUNAM, http://www.iingen.unam.mx, desde que se terminó de editar.

Abstract

The problem of automatic fault diagnosis and tolerant control for dynamic systems has become essential research topic during the last years as part of the industry's monitoring and supervision of complex systems. This fact has motivated diverse research groups who are developing fault diagnosis technologies in Mexican academic institutions. Thereby, the fault diagnosis community considered it relevant to share our experience by showing the benefits of the novel diagnostic techniques implemented by software with some applications tested in our country.

Thus, this monograph, written by several experts, has been prepared by selecting the main topics discussed during the seminar on Fault Diagnosis and Tolerant Control organized face-to-face and virtual by the authors over the last four years. The Topics features include: (a) The design of a robust computer numerical control CNC of new generation; (b) The faults isolation problem in sensors and actuators of a helicopter prototype; (c) Failure diagnosis in the boiler system of a distillation column; (d) The remaining functional live analysis of an aerospace propulsion system; and (e) Three issues associated with leaks in pipelines. One relates to a water distribution network; the second shows an algorithm to isolate sequential leaks in the main conduit, and the last links with the estimation of the head loss of the line in real-time.

Keywords: Model-based fault diagnosis, Data-driven fault detection, Degradation model, Inteligent Control System.

This page intentionally left blank

Resumen

El problema del diagnóstico automático de fallas y control tolerante para sistemas dinámicos ha adquirido cierta importancia como parte de la tecnología de monitoreo y supervisión de sistemas complejos. Este interés se ve reflejado en la creación de diversos grupos de investigación en instituciones académicas de México que cultivan la problemática. De este modo, la comunidad de diagnóstico de fallas consideró relevante compartir nuestra experiencia mostrando los beneficios de algunas técnicas novedosas de diagnóstico implementadas por software con aplicaciones validadas en nuestro país.

Así, esta monografía redactada por diversos expertos, fue preparada seleccionando los temas principales discutidos durante el seminario de Diagnóstico de Fallas y Control Tolerante llevado a cabo de manera presencial y virtual durante los últimos cuatro años por los expertos. Los temas tratados incluyen: (a) El diseño de un control numérico por computadora CNC; (b) El problema de aislamiento de fallas en los sensores y actuadores de una maqueta de un helicóptero; (c) El diagnóstico de fallas en el sistema de calentamiento de una columna de destilación piloto; (d) El análisis de la vida funcionales de un sistema de propulsión aeroespacial; y (e) Tres problemas asociados con fugas en ductos. Uno relacionado con la localización de fugas en una red de distribución de aguas; el segundo, muestra un algoritmo que opera en tiempo real para aislar fugas secuenciales en un ducto principal; y el último relacionado con la estimación en tiempo real de la perdida de carga de una línea.

Palabras claves: Diagnóstico de fallas basado en modelo, Detección de fallas basada en datos, Modelo de degradación, Sistema de control inteligente.

This page intentionally left blank

Preface

The high quality and ramifications of the production processes, the complexity of technological infrastructure, and software systems currently demand the design of reliable and efficient integrated automatic control systems. It is well known that a single fault can develop into multiple defects if the operator ignores or misunderstands the warning signals. These reasons have motivated the development of advanced automatization technologies by including supervision and monitoring tasks with fault detection and diagnosis facilities. This implies the design and implementation of automated devices based on physical principles, knowledge, and data-driven techniques. As a result of the aforementioned current demand, multidisciplinary diverse groups have been integrated that design diagnostic tools for specific applications by considering control and estimation theory, artificial intelligence tools, and engineering knowledge of the critical physical system in question. This development has led to the crucial concepts associated with fault diagnosis during the last 40 years. Advanced tools have been conceived in cooperation with researchers, engineers and technicians concerned about the safety, reliability and health performance of physical systems for society's benefit. Thus, critical fault scenarios for each engineering field have been studied and the fields' diagnosis tools have been implemented. Vulnerable industries such as aeronautic, nuclear, transportation, petrochemical, and electrical are examples where real-time fault diagnosis requirements are mandatory. By considering wireless communications and digital devices, one can comment that the advanced supervision systems with fault diagnosis allow improvements in aircraft maintenance, complex processes, rotating machines, autonomous vehicles, and service infrastructures, such as water, electricity and wireless communication. As a consequence, this technology can be extended to all physical systems, and it produces better environmental protection, more effective maintenance schedules, and better product quality.

The main objective of this monograph is to provide a set of technical contributions related to the critical subject of safe automatic control systems. In particular, advanced fault detection and isolation solutions, as well as fault-tolerant controls, are addressed, and some applications exemplify the advantages of the safe control technology implemented by software. The topics of this manuscript related to safe control have been studied by academic colleagues and postgraduate students of 10 institutions who have regularly attended the workshop Fault Diagnosis and Tolerant Control from the Spanish *Diagnóstico de Fallas y Control Tolerante* initiated and organized by academics over the last four years. Based on the material introduced in the workshop, some applications were selected that will be presented and discussed by using experimental results. Thus, this project is the result of the academic relationship between authors, and part of the content was generated from much fruitful discussions within the work-

shop's diagnostic and monitoring group and invited academic colleagues. The title of the monograph, **Fault Diagnosis and Tolerant Control: Applications**, was chosen to reflect the broad scope of the current fault diagnosis methods and the experience of our community. Seven case studies and their solutions were selected, where some fault diagnosis and control techniques will be discussed by using experimental results. The authors are all specialists in their subject, and the distribution of the chapters was selected based on their experience.

The presentation of the topics is accessible but formal, and a reader with a background in automatic control systems and signal processing can easily follow it. Within this monograph, all the procedures are introduced by considering the real cases, and each chapter can be read independently. In this way, the project also indirectly strives to promote the advantages of some feasible applications where advanced fault diagnosis tools and the fault-tolerant control can be used to improve the safety and reliability of critical physical systems.

This project was possible through the effort of the authors to present their recent research topics and to disseminate the newly available technologies for the improvement of security and system reliability. I thank all the authors for their support and dedication while preparing the manuscript. On behalf of all of them, I will like to thank our institutions that gave us the opportunity to prepare the text under pandemic conditions:

- Instituto Tecnológico de Morelia,
- Instituto Tecnológico de Tuxtla Gutiérrez,
- Tecnológico de Monterrey,
- Universidad Tecnológica de La Habana José Antonio Echeverría,
- Universidad Autónoma de Nuevo León,
- Vanderbilt University,
- Universidad Michoacana de San Nicolás de Hidalgo, and
- Universidad Nacional Autónoma de México.

I will also wish to thank the reviewers of the Comité Editorial of the Instituto de Ingeniería, who helped us improve the manuscript, and Biologist Blanca Gamboa and the proofreader Mary-Ann Hall for their support in complying with the editing and typography standards of the manuscript.

Mexico City

Cristina Verde
February 2023

List of Contributors

Efraín Alcorta García
Universidad Autónoma de Nuevo León, Facultad de Ingeniería Mecánica y Eléctrica, Mexico,
e-mail: efrain.alcortagr@uanl.edu.mx

Juan Anzurez Marin
Universidad Michoacana de San Nicolás Hidalgo, Mexico, e-mail: juan.anzurez@umich.mx

Rolando Carrera
Universidad Nacional Autónoma de México, Mexico, e-mail: rcarrera@unam.mx

Mario Heras Cervantes
Tecnológico Nacional de México/Instituto Tecnológico de Morelia, Av. Tecnológico 1500, Col. Lomas de Santiaguito, Morelia, Mexico e-mail: mario.hc@morelia.tecnm.mx

Francisco-Ronaly López-Estrada
Instituto Tecnológico de Tuxtla Gutiérrez, Mexico e-mail: frlopez@tuxtla.tecm.mx

Orestes Llanes-Santiago,
Universidad Tecnológica de La Habana José Antonio Echeverría, CUJAE, Marianao, La Habana, Cuba,
e-mail: albprieto@automatica.cujae.edu.cu

Ruben Morales-Menendez
Tecnológico de Monterrey, México e-mail: rmm@tec.mx

Aldo J. Muñoz Vázquez
Texas A&M University, Department of Multidisciplinay Engineering, McAllen, TX 78504, USA,
e-mail: aldo.munoz.vazquez@gmail.com

Vicenç Puig
Universitat Politécnica de Catalunya, Spain, e-mail: vicenc.puig@upc.edu

Alberto Prieto Moreno
Universidad Tecnológica de La Habana José Antonio Echeverría, CUJAE, Marianao, La Habana, Cuba,
e-mail: albprieto@automatica.cujae.edu.cu

Marcos Quiñones-Grueiro
Institute for Software Integrated Systems, Vanderbilt University, Nashville, Tennessee, USA,
e-mail: marcos.quinones.grueiro@vanderbilt.edu

Luis Alejandro Ramírez
Universidad Autónoma de Tamaulipas, Unidad Académica Multidisciplinaria Reynosa Rodhe, Mexico,
e-mail: laramirez@docentes.uat.edu.mx

Ricardo A. Ramírez Mendoza
Tecnológico de Monterrey, México, e-mail: ricardo.ramirez@tec.mx

Gerardo Romero Galvan
Universidad Autónoma de Tamaulipas, Unidad Académica Multidisciplinaria Reynosa Rodhe, Mexico,
e-mail: gromero@docentes.uat.edu.mx

Ildeberto Santos-Ruiz
Instituto Tecnológico de Tuxtla Gutiérrez, Mexico, e-mail: ildeberto.dr@tuxtla.tecm.mx

Adriana del Carmen Tellez Anguiano
Instituto Tecnológico Regional de Morelia, Mexico, e-mail: adrianat@itmorelia.edu.mx

Lizeth Torres
Instituto de Ingeniería, Universidad Nacional Autónoma de México, Mexico,
e-mail: FTorresO@iingen.unam.mx

Guillermo Valencia-Palomo
Tecnológico Nacional de México, IT Hermosillo, México, e-mail: gvalencia@hermosillo.tecnm.mx

Antonio Jr. Vallejo Guevara
Tecnológico de Monterrey, Mexico, e-mail: avallejo@tec.mx

Cristina Verde
Instituto de Ingeniería, Universidad Nacional Autónoma de México, México, e-mail: verde@unam.mx

Adriana Villalón Falcón
Universidad Tecnológica de La Habana José Antonio Echeverría, CUJAE, La Habana, Cuba
e-mail: avillalon@automatica.cujae.edu.cu

Manuel Alejandro Zuñiga
Universidad Autónoma de Tamaulipas, Unidad Académica Reynosa Rohde, Mexico,
e-mail: manuel.zuniga@docentes.uat.edu.mx

Contents

This page intentionally left blank

Chapter 1
Introduction

Cristina Verde[1]

Abstract This chapter serves as an introduction to the main topic of this monograph related to some applications of advanced technologies for fault diagnosis and tolerant control by using the concepts of redundancy and estimation for qualitative and quantitative models. These technologies implemented mainly by software operate in real-time and increase the security of infrastructures and indirect environmental protection. These tools also constitute a qualitative step in analyzing the behavior of complex systems, their operation in failure conditions, and the stage of a safe stop in an emergency condition. At the initiative of various academics in the country, who cultivate the area of security and fault-tolerant control, a series of seminars were organized to reinforce the area and encourage young engineers to work in it. The original idea was to rotate the event between various institutions. The pandemic forced us to change the seminar to a virtual form. In September 2021, the first cycle of the meetings was concluded, generating this monograph with some projects discussed in the forum. The chapter includes a summary of the different applications presented in the monograph and discussed throughout the set of seminars **Fault Detection and Isolation, and Tolerant Control** performed with the participation of academic experts in the field and postgraduate students of our institutions over the last four years.

1.1 Background and motivation

Physical systems and their operation have become more complex in the last forty years since the interconnection of components and devices have increased, and the physical processes involved in the systems are now more sophisticated (Clark (1978); Patton et al. (1989); Escobet (2019)). Thus, society demands a reliable operation of critical systems such as hazardous fluids and electricity transportation networks. In this century, the platform explosion of the British Petroleum Company in the Gulf of Mexico caused the largest marine oil spill and is an example of the incidents that should not occur. Thus, governments and international institutions have established regulatory security rules. The implementation of these rules, however, only reflects reality in some industries, services, and societies. Production and profitability are considered in many cases the highest priority in a system instead of safety warnings. Therefore, it is our responsibility as engineers and technicians to assume the worst possible fault scenarios in system

[1] Instituto de Ingeniería UNAM, Mexico, e-mail: verde@unam.mx

adjustment to avoid damaging the society and the environment through abnormal events in systems and infrastructure.

Technology can be considered valuable as long as it can provide a service to society in a reliable way. Therefore, society is demanding that safe technologies become more reliable and safe in the operation of any component of a physical system. This requirement extends beyond nuclear reactors, aircraft, and many chemical processes to systems likes autonomous vehicles, power plants, hydraulic networks, and road networks, where availability is vital. We emphasize that the reliability specification is a crucial concern in the design of digital devices and modern infrastructure depending on advanced technologies, also known as **intelligent systems**. Reliability can be defined as the ability of a system to perform its intended function over a given period of time (Blanke et al., 2006). Incidents and disasters that are linked to the absence of reliability are failure states. This means the inability to perform the intended function of a system. This condition can be the effect of a fault which is a change in the behavior of a system, or part of it, from the behavior that was set at design time. This implies that everything that cannot be accounted for as the effects of a fault should be blamed on an incorrect design, which appears to be an overall cause of failures. By considering the temporal dependency of a fault, three classifications have been proposed by the safe process community: an abrupt or stepwise fault, an incipient or drift-like fault, and an intermittent or random fault (Blanke et al., 2006).

Faults can occur in the proper self system or in the accessory devices, as are sensors, actuators, or protection systems. The fault's origin can be diverse, including human error. On the other hand, by considering that automatic control systems comprehensively integrate information about processes, it is understandable that this discipline began in the eighties to design supervision systems through software (Himmelblau, 1978). The International Labor Organization, an agency of the United Nations in charge of labor problems, reports that 70% of industrial accidents are caused by human error.

Considering that process safety covers various fields of knowledge, from its origins, the technical committee T6.4 SAFEPROCESS of the International Federation of Automatic Control made efforts to standardize the terminology used in the publications and achieve common languages independent of the application. This fact motivated the creation of the terms: Fault Detection, Fault Isolation, Fault Diagnosis, Fault Tolerant Control, and others. In the Glossary of this monograph, readers can consult the relevant definitions taken from the book by Blanke et al. (2006).

To summarize, the control theory, the development of artificial intelligence, and the low cost of digital devices motivated the search for automatic diagnosis techniques for critical systems (Chow and Willsky, 1984; Isermann, 1984; Frank, 1990). This allows the possibility of developing **fault detection and isolation** (FDI) algorithms with a low cost for diverse applications, such a the automotive industry, medical devices, and household goods. According to Venkatasubramanian et al. (2003), three large groups of FDI methods emerged based on historical data and qualitative or quantitative models. On the other hand, artificial intelligence is now another field that is very active in diagnostic problems, which has enriched the discussions and the solution options for the diagnosis problems. Reiter (1987) and de Kleer and Kurien (2003) tackled the problem from a logical reasoning perspective, and the failure events were rated as suspect or declared. The reader interested in a detailed historical evolution of the safe process can consult the references given in Verde et al. (2013).

One possibility in diagnosing the system state and improving security is to use **physical redundancy**: to double or triple devices. In addition, by vote, one detects the faulty component. This solution is costly. By replacing hardware with models and algorithms, diagnosing the state of the elements, is preferred.

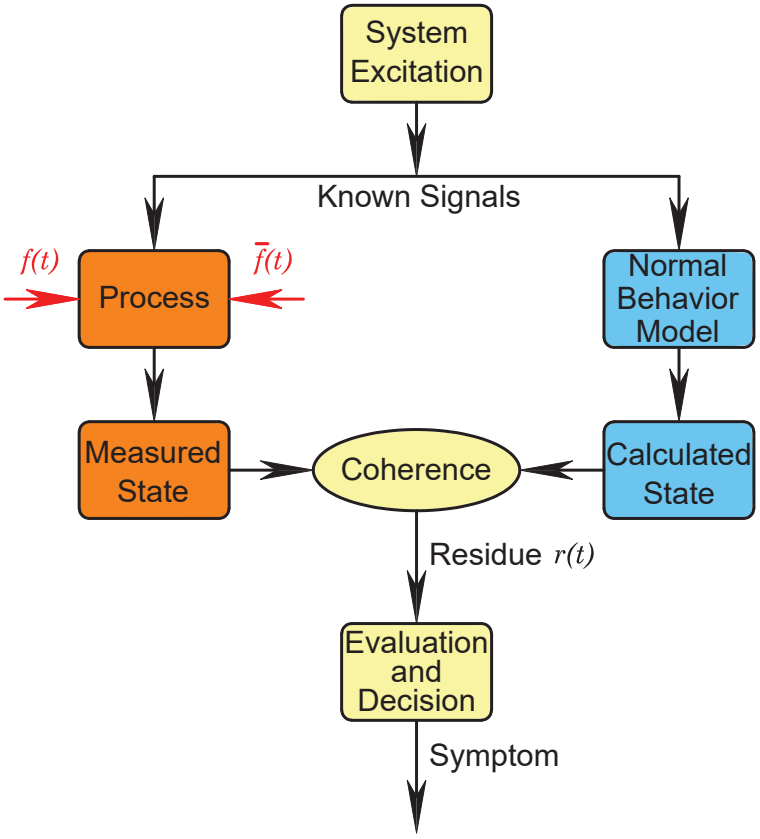


Fig. 1.1: Coherence principle where fault $f(t)$ and perturbation $\bar{f}(t)$ affect the process and the model disregards both signals

This type of diagnosis is denoted **analytical redundancy** to distinguish it from the physical case. This concept can be generalized to **a redundant structure by software** if two different ways to describe the system with similar behaviors in normal conditions exist. One form corresponds to physical system measurements and the other to the statement of the health system under study. Thus, both descriptions are compatible with health conditions, and, if a fault is present in the physical system, both descriptions are incoherent. Thus, the structure of the redundancy principle shown in Fig. 1.1 is the central key of a diagnosis task, and the model could be qualitative, quantitative or a data set. It is important to remark that in some applications, physical redundancy continues to be a security requirement formally legislated by international agreements as in aeronautical applications, where five or six computers coexist with independent power sources for each actuator and many redundant sensors from different manufacturers (Goupil, 2009; Muenchhof et al., 2009).

Maintaining a high level of safety, performance and reliability is essential for detecting prompt system errors, component faults, abnormal operation and unexpected events (Verde et al., 2013). This fact includes the knowledge of the cause and its severity, so the operator together with the supervisory control and data acquisition (SCADA) system can take corrective actions. Thus, for economic and social reasons, it is necessary to improve and propose control technologies. This novel technology named **fault-tolerant control** (FTC) allows reducing incidents, accidents, and the suspension of critical services. By consid-

ering that the common objective of an automatic control system is to satisfy good performance and efficiency, measured by the quality and quantity of the product, and if safety specifications are included in the operation of the system, it now becomes intelligent.

Therefore, innovative FTC algorithms must correct errors produced by a fault. This work could be done through self-control if it is robust regarding the fault or with a loop modification. The change could be by rescheduling the control loop or by process reconfiguration. In the worst case, the operator can decide to shut down the system, for safety reasons.

Consequently, the fault detection and isolation, together with tolerant control in real-time are significant requirements of current control technologies, which must alert operators and managers as soon as possible about deteriorated, abnormal conditions and at the same time hold a safe control. This new paradigm can be called resilient control because it has the ability to recover from fault effects.

Cyber-physical systems are impacted our society, leading to the establishment of new services in diverse applications. This type of system can be considered a new generation of digital technologies involved in the integration and inter-dependencies of the cyber and physical world as a combination of facilities, equipment, procedures, communications, and computational elements to provide services. These systems have generated new challenges in the field of safety systems, including fault and attack tolerant control. The reader interesting in this specific topic can read the survey written by Cao et al. (2020).

1.2 Fault detection and tolerant control

1.2.1 On-line supervision

Advanced integrated SCADA systems aim to operate the global process with satisfactory performance in nominal conditions and in its diverse operating modes, including start, stop and fault conditions. The traditional mechanism for fighting the fault modes is through protection, such as fuses, floats, one-way valves, and handbrakes. This form is valid only for short faults and the avoidance of damage propagation.

Since possible fault effects can range from total loss or stoppage of a process to a simple operation point modification, the fault diagnosis and its management is a challenging task. For example, a steel mill outage costs more than 80000 USD per hour, which justifies the effort to integrate diagnostic tasks with the fault-tolerant control. Figure 1.2 shows the organization with the diverse functions that integrate an advanced supervision system. The supervision is performed through a set of tasks that can be executed asynchronously depending on the current situation. For industrial conditions, the design of a fully automated supervisor is a challenging task and requires well-known process behavior. Moreover, one must remember that a complex system has hundreds of variables that overload operators and pilots, produce stress and lead to incorrect decisions. This condition is known as a cognitive overload (Bainbridge, 1983). Today, the implementation of a supervisory system through edge computing networks and clouds enables a reduction of the operator workload.

The **alarms** in the diagnosis framework are considered discrete events activated if a variable or function exceeds certain limits. This is the usual information for a monitoring application. Figure 1.3 allows us to explain the alarm activation as a function of thresholds that delimit the acceptable value set for each

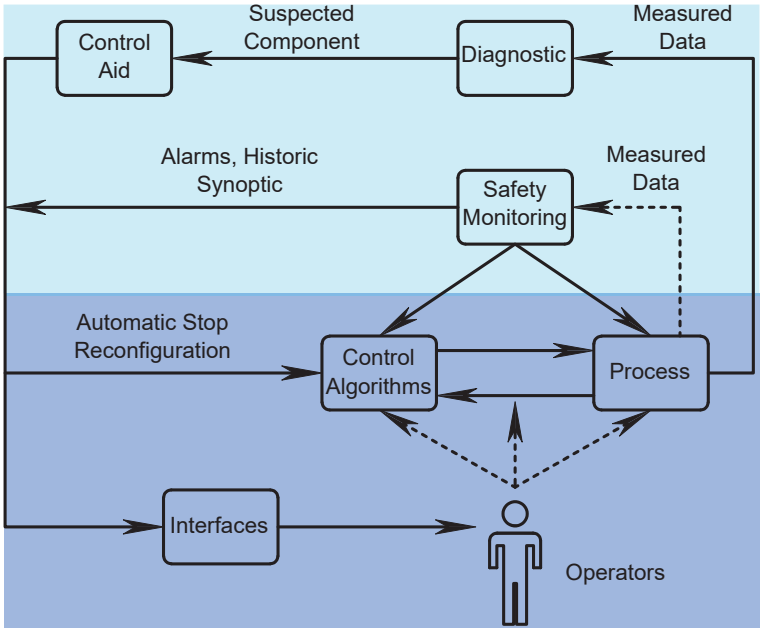


Fig. 1.2: Monitoring scheme with advance supervision tasks

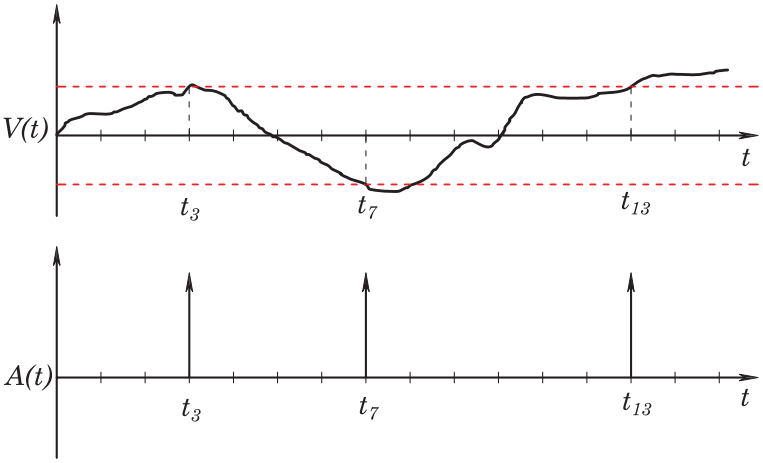


Fig. 1.3: Top graphic: Time evolution of the variable $V(t)$ where the red lines indicate thresholds. Bottom graphic: discontinuous signals denoted $A(t)$ associated with the alarm impulse of $V(t)$

variable. The system usually operates in normal conditions as long as the variable $V(t)$ is between the thresholds, marked by dotted red lines. When, it exceeds them, however, either by an overly low or high value, an alarm is generated, which is shown at the bottom of the figures as pulses that are denoted by $A(t)$.

1.2.2 Diagnosis process

The diagnosis process is frequently performed in stages. In general, the inputs of the diagnosis algorithm are the measurements of the system under study, and two families of diagnostic systems are available. Figure 1.4 describes the two main procedures for the design of a FDI algorithm a) indicates the steps for the methods based on mathematical models of the systems and the track b) displays the process for signals and data-driven methods. In a practical application, the engineer in charge of the supervisor design selects the appropriate one, according to the system knowledge, the fault modes, and the available sensors. For sequence a), a fault is first detected from the incoherence of the mathematical model's incoherence and data. Thus, the detection signals called **residuals** allow the symptom generation, and the isolation or localization of the fault is achieved from the features of the symptom. This stage concludes when the faulty element is identified and how it affects the system. For the option of signals and data-driven methods, a set of experiments and historical data are collected such that the designer can determine relevant features and attributes of the faults, for instance statistical parameters. These parameters must be appropriate for the classification or recognition of the faults. This stage is critical for achieving satisfactory performance in the diagnosis. The last part of the stage is the fault mode classifications, and diverse tools from the artificial intelligence field can be used for this purpose.

For both types of diagnosis procedures briefly described before, one assumes that the control system is properly designed and built. Later, the behavior deviates from the nominal and normal conditions caused by poor maintenance, natural events, exceeded longevity, or sabotage. The term fault-tolerant or robust system is understood as a system that can cope with some faults while providing its service or at least a somewhat degraded version of its functionality.

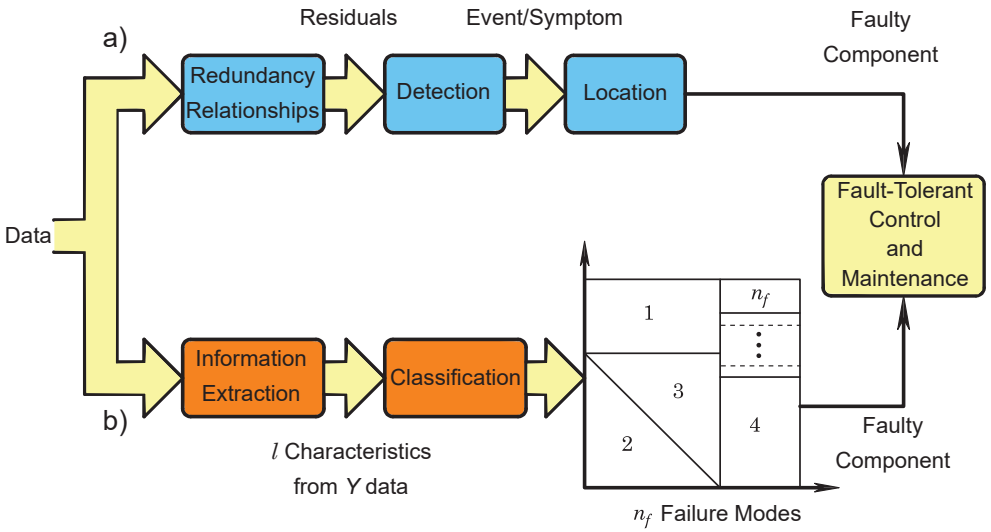


Fig. 1.4: Diagnosis stages from process measurements by using mathematical, signal, or data-driven models

1.3 Structure and monograph content

Specifically, this monograph focuses on methods and applications for detecting, isolating, and identifying changes in the behavior of dynamical systems and controlled processes. The introduced tools are based on qualitative and quantitative models, historical data, and rule-based information. Thus, the book's content will be a helpful reference for newcomers to the applications of fault detection and isolation and fault-tolerant control (FDI-FTC) as well as for those engineers and researchers interested in available new tools developed for improving the safety of dynamic systems.

Chapter 2 addresses the design of a control system for the new generation of computer numerical control (CNC). The main contribution is the system optimization based on the Markov Decision Process (MDP). The process consists of four modules, namely: data acquisition, cutting tools, surface roughness, and planning. Thus, the designed intelligent control guides the actions of the operator in peripheral milling processes. Practical results show the performance for the main parameter *Ra*, which corresponds to the surface roughness.

For academic and teaching purposes, Chapter 3 deals with the fault diagnosis technology by using generalized Hamiltonian models. Specifically, the problem of fault isolation in actuators and sensors is solved with a bank of observers for a nonlinear 2-DOF helicopter scale Hamiltonian nonlinear model. The whole diagnosis program is implemented to be tested as a real-time embedded system. Experimental results show the effects of sampling time, control and fault sensitivity. The diagnosis system is implemented at the automatic control laboratory of the Universidad Autónoma de Nuevo León.

By considering the EDF-1000 distillation pilot plant, Chapter 4 presents a fault detection algorithm for a boiler system of a column described by a Takagi-Sugeno fuzzy model. For the residual generation, sliding mode observers are used. Specifically, the considered faults are part of the buck-boost electric converter, which regulates the power through the duty cycle of the power electronics. The chapter ends with some experimental results where load fault, input voltage variations, and a combination of defects are detected through the residuals.

Chapter 5 examines the problem of the remaining useful life (RUL) prediction of physical infrastructure. This new issue is motivated by the practical necessities for establishing maintenance policies strategies since the RUL features are time varying. In particular, a novel adaptive prognosis method is introduced by using a simple performance metric. The measure periodically considers the RUL prediction and maintenance planning. The procedure is applied to a commercial simulator of an aerospace-propulsion system, and the results show satisfactory performance.

The last three chapters address diverse problems, all related to the leak detection and localization in pipelines. In particular, Chapter 6 proposes a leak localization method for water distribution networks using k-NN classifiers with a nonlinearly transformed pressure residual as a feature. The main property of the transformation is the residual direction extraction, separating it from its magnitudes. Simulated results with EPANET as shown for the Hanoi network and also physical data from the Madrid network are used to test the procedure.

Since the leak diagnosis systems in pipelines are included as supplements of the actual SCADA systems, Chapter 7 describes a smart SCADA system that provides for as part of its tasks a real-time algorithm for the sequential leaks location of a 162 m prototype hydraulic pipeline. The prototype is instrumented with upstream and downstream pressure sensors to regulate the operation of the line un-

der diverse operation conditions. The hardware and software infrastructure is designed specifically at UNAM as a test bank for the study and design of fault diagnosis technologies. Note that the SCADA system with the embedded fault diagnosis system is implemented in LabVIEW and the results are captured in real-time by using an industrial portable PC.

The success of a leak diagnosis is strongly dependent on the system model that could not be appropriated for computational implementation tasks, and the models frequently involved uncertain parameters. Chapter 8 addresses the problem of looking for computational models that can be useful for fault diagnosis and tolerant control and, at the same time simplifying the estimation online of critical parameters. For example, a real-time identification procedure is presented, which allows the estimation of the critical parameters of a power law model for the head loss in pipelines. The identification task is achieved by a state observer where the state consists in a combination of physical parameters of the line. Experimental data from the hydraulic test bank introduced in Chapter 7 validated the proposed observer. Moreover, a simple statistical study with simulated data shows the effect of an uncertainty parameter of the model in the probability density function of a leak location.

References

Bainbridge, P. (1983). Ironies of automation. *Automatica*, 6:775–779.

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2006). *Diagnosis and Fault Tolerant Control*. 2nd. Ed. Springer, Berlin.

Cao, L., Jiang, X., Zhao, Y., Wang, S., You, D., and Xu, X. (2020). A Survey of Network Attacks on Cyber-Physical Systems. *IEEE-Access*, 8: 44219–44227.

Chow, E. and Willsky, A. (1984). Analytical redundancy and the design of robust failure detection systems. *IEEE-Trans. Automatic Control*, 29(7):603–614.

Clark, R. N. (1978). Instrument fault detection. *IEEE Trans. on Aeroesp. and Elec. Sys.*, AES-14(3):456–465.

de Kleer, J. and Kurien, J. (2003). Fundamentals of model-based diagnosis. In *IFAC Symposium on Fault Detection, Supervision and Safety of Technical Process Safeprocess, Washington, DC*.

Escobet, T. e. a. (2019). *Fault Diagnosis of Dynamic Systems*. Springer, Berlin.

Frank, P. (1990). Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy. *Automatica*, 26(2):459–474.

Goupil, P. (2009). AIRBUS state of the art and practices on fdi and ftc. In *7th IFAC International Symposium on Fault Detection, Supervision and Safety of Techinical Processes, Barcelona*.

Himmelblau, D. M. (1978). *Fault Detection and Diagnosis in Chemical and Petrochemical Processes*. Elsevier. Chemical Eng. Monograph 8.

Isermann, R. (1984). Process fault detection based on modeling and estimation method - a survey. *Automatica*, 20(4):387–404.

Muenchhof, M., Beck, M., and Isermann, R. (2009). Fault tolerant actuators and drives-structures, fault detection principles and applications. In *7th Ifac International Symposium on Fault Detection, Supervision and safety of Technical Processes*.

Patton, R. J., Frank, P. M., and Clark, R. N. (1989). *Fault diagnosis in dynamic systems: Theory and applications*. Prentice Hall, (eds).

Reiter, T. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95.

Venkatasubramanian, V., Rengaswamyd, R., Yin, R., and Kavuri, S. (2003). A review of process fault detection and diagnosis; part i: Quantitative model based methods; part ii: Qualitative model and search strategies; part iii: Process history based methods. *Computers and Chemical Engineering*, 27:293–346.

Verde, C., Gentil, S., and Morales-Menéndez, R. (2013). *Monitoreo Y Diagnóstico Automático De Fallas En Sistemas Dinámicos*. Trillas.

Chapter 2
Intelligent Control System for High Speed Machining

Antonio Jr. Vallejo Guevara[1], Ricardo A. Ramírez Mendoza[1] and Ruben Morales-Menendez[1]

Abstract The next-generation of high-speed machining (*HSM*) systems demand advanced features such as intelligent control under uncertainty. This requires, in turn, an efficient administration and optimization of all system's resources towards a previously identified objective. This research project presents an optimization system based on Markov decision process (*MDP*), where an intelligent control guides the actions of the operator in peripheral milling processes. The system bases its decisions on prediction models of the machining process: state of wear of the cutting tool and surface roughness; therefore, an estimator based on multi-sensors and data fusion will be analyzed. Exploiting intensive experimentation, databases, models, and detailed procedures generates high precision models. The initial results and the continuous development based on machine learning (*ML*) for *MDP* demonstrate great expectations of this proposal.
Keywords: Markov decision, Process optimization, High speed machining, Milling process, Artificial neural networks

2.1 Introduction

HSM requires high magnitudes of spindle speeds, feed rates, as well as high acceleration and deceleration rates. Simultaneously, it is subjected to stringent restrictions such as low machining cost and time, and high precision and accuracy. Intelligent machines can meet the strong competitiveness demanded by new businesses. Intelligent *CNC* machines convey advanced features such as prediction of operations, reduction of setup time, detection of cutting tool condition, acquisition of knowledge, and inferences from incomplete information Balic (2006). However, process planners still have great difficulties for measuring *on-line* process data on machining processes such as cutting tool wear condition (*CTWC*) and surface roughness (*Ra*) Jawahir and Wang (2007). An intelligent control system for *HSM* is described, exhibiting several desirable features such as: prediction of key variables (*Ra* and *CTWC*), definition and adaptation of optimal cutting conditions and operation policy, and an objective function-based optimization. Special emphasis is placed on *Ra* and *CTWC* modeling due to the resurgence of some *ML* algorithms, the Industry 4.0 and the industrial Internet of things (*IIoT*). A novel mathematical framework for fault diagnosis

[1]Tecnológico de Monterrey, Av. E. Garza Sada # 2501 , e-mail: avallejo@tec.mx, ricardo.ramirez@tec.mx and rmm@tec.mx

(i.e. cutting tool wear condition and surface roughness) is proposed as a key support of intelligent control under uncertainty for *HSM* systems.

2.2 State of the Art

2.2.1 High-Speed Machining

Several optimization methods have been developed around process planning systems for machining processes. For example, a procedure for tool selection in milling operations was proposed in Carpenter and Maropoulos (2000). First, several alternatives of cutting tools were considered by an iterative method. Then, cutting data was refined by a set of technological constraints, including tool life, surface finishing, machine power, and available spindle speeds and feed rates. Finally, three user-defined optimization strategies were available (minimum cost, maximum production rate, or predefined tool life).

In Dereli et al. (2001), a cutting parameters optimization system, based on a two-stage methodology, was introduced. First, a tentative number of passes and depth of cuts were determined through the so-called volume sectioning method. Then, the cutting speed and feed rate for each pass were optimized by using genetic algorithms (*GA*). A second-order mathematical model was developed for predicting the parameter *Ra* as a function of the cutting speed, feed rate, depth of cut, and nose radius of the cutting tool in turning operations Suresh et al. (2002).

Based on previous work by Dereli et al. (2001), an algorithm for the selection of optimal cutting conditions was proposed in Mursec and Cus (2003), allowing the calculation of several cuts required and machining time. Zuperl et al. (2004) presented a new hybrid optimization technique based on the maximum production rate criterion and ten technological constraints. A general algorithm, called *OPTIS*, was used in conjunction with artificial neural networks (*ANN*) to solve the complex optimization problem. *OPTIS* selects the optimum cutting conditions (based on minimum machining costs) from commercial databases. *ANN* ensured an efficient and fast selection of the optimum cutting conditions and processing of available technological data. Compared to the *GA* and linear programming approaches, this hybrid optimization technique improved the optimal cutting parameters selection by around 30.41% and 20%, respectively. Based on *OPTIS*, Zuperl et al. (2006) proposed an adaptive neural controller for on-line optimal control of a milling process.

A two-phase optimization strategy based on the *Taguchi* dynamic characteristic theory was proposed in Tzeng and Chen (2005). Experimental results showed that the machining time could be reduced with low process variance and increased robustness of the *CNC* milling processes.

Yih-Fong (2005) presented a *Taguchi* method coupled with principal component analysis for the optimization of *HSM* milling processes. Optimal process conditions were selected for producing the best dimensional precision and accuracy, *Ra*, and *CTWC*. The selected control factors were: milling type, cutting speed, feed per tooth, film material, tool material, number of teeth, rake angle, and helix angle. Based on the cutting parameters optimization system an index for the inter-correlated multiple performance features of a *HSM* milling process was computed, obtaining optimized settings.

A genetically optimized neural network system that selects the optimal cutting conditions for milling processes was proposed by Tansel et al. (2006). *GA* was used to maximize the rate of metal removal and

minimize the *Ra* based on different *ANN* models. A mathematical model based on both material behavior and machine dynamics were described in Palanisamy et al. (2007).

Mukherjee and Ray (2006) reviewed different optimization techniques in metal cutting processes, discussing a general framework. Reviewed optimization methods currently applied were: *Taguchi* method, response surface methodology (*RSM*), *Mathematical Iterative Search Algorithm*, *GA*, and simulated annealing methods.

In Kant and Sangwan (2014) the authors presented a multi-objective predictive model for the minimization of power consumption and *Ra* during the machining of *AISI 1045* steel. The predictive model used grey relational analysis coupled with principal component analysis and response surface methodology *RSM*. The statistical significance of the proposed predictive model has been tested by the analysis of variance. The obtained results indicate that feed is the most significant parameter followed by the depth of cut and cutting speed to reduce power consumption and *Ra*. The results provide an improvement of 6.59% in power consumption and 2.65% in *Ra* over the best experimental run.

In Karabulut (2015) milling tests were performed based on the *Taguchi DoE* method using *L18* orthogonal array and determined the optimal cutting parameters for *Ra* and cutting force. The analysis results showed that material structure was the most effective factor on *Ra* and feed rate was the dominant factor affecting cutting force. *Ra* values were significantly improved by between 196% and 312% in milling Al_2O_3 particle-reinforced *Aluminum* alloy composite compared to *AA7039* Aluminum. *ANN* was able to predict the *Ra* and cutting force with a *MSE* equal to 2.25% and 6.66%.

In Karkalos et al. (2016) models for the prediction of the *Ra* in milling of *Ti-6Al-4V ELI Titanium* alloy were developed. A set of experiments based on *Box Behnken Design* for a three factor and three level central composite design concept was conducted. Depth of cut, cutting speed and feed rate were selected as input parameters and *Ra* was measured as output. The *MSE* and the correlation coefficient were determined to be 8.6×10^{-3} and $R^2 = 0.97$, respectively; for *ANN* models were 2×10^{-3} and $R^2 = 0.98$, for the test group of the optimum model. Almost all of the previous works are usable within narrow operating conditions only.

In Yang et al. (2019), the authors developed an integrated prediction model based on trajectory similarity and support vector regression, which can predict the tool wear and life. The time domain and wavelet analysis are carried out. Five eigenvectors were selected as the input vectors of the prediction model by studying the correlation between 45 characteristic quantities and the tool wear. The relative errors of flank wear value prediction accuracy in the stable stage of the sample tool were above 88% and the prediction accuracy of the stable stage of Tool 1, 2, and 3 was 88.5%, 87.5%, and 90.5% respectively, by using this integrated prediction model. In Hoang et al. (2020), the authors presented an intelligent control system for self-adjusting on-line cutting condition for *HSM* by considering the tool-wear amount to keep the machined product's quality in allowable limit. The empirical analysis of variance and *ANN* were used. The analysis of variance was used for generating the empirical functions which were used as the boundary condition as well as constraint evaluation. The *ANN* was used for generating the new optimal cutting condition.

2.2.2 Surface Roughness (Ra) Modeling

Surface roughness (*Ra*) has been subject of research for years where four major approaches have been pursued Benardos and Vosniakos (2003): (i) Machining theory; (ii) Experimental investigation; (iii) Design of experiments (*DoE*); and (iv) Artificial intelligence (*AI*), a brief review of these is presented next.

Machining theory. It considers geometric and analytical modeling. Geometric modeling is based on the geometrical motion of a metal cutting process regardless of the cutting dynamics. Some geometric models for turning operation are shown in Boothroyd and Knight (2006). In Lee et al. (2001), a geometric model and a simulation algorithm were developed, calculating cutting parameters, cutter and work-piece geometry, vibration signals, and run-out parameters. Finally, Sai and Bouzid. (2005) used an experimental design with two factors (cutting speed and feed per tooth) and five levels to validate the *Ra* model.

Experimental investigation approach. It requires a full range of experiments handling the most relevant factors, and the results are used to investigate the effect on the observed quality characteristic. In Barber et al. (2001), the *Ra* was computed as a quadratic function of tool service time, and the major parameters involved were tool geometry and *CTWC*. In Abouelatta and Madl (2001), mathematical models for predicting *Ra* were derived based on cutting parameters and vibration signals.

Design of Experiments (*DoE*) approach. The *RSM* and *Taguchi* techniques for *DoE* are the most widespread methodologies for *Ra* prediction. In Ozcelik and Bayramoglu (2006), a statistical model was used for estimating *Ra* in high-speed flat milling, where the parameters involved were the total machining time, depth of cut, step over, spindle speed and feed rate. Similar works are presented in Abouelatta and Madl (2001), Özel and Karpat (2005) and Morales-Menendez et al. (2005). In Subramanian et al. (2014) a mathematical model was developed to predict the *Ra* in terms of machining parameters such as radial rake angle, nose radius of cutting tool, spindle speed, feed rate and axial depth of cut. The authors applied the *RSM* to define the *DoE*; the *Ra* was computed with a second-order quadratic model, which was obtained by applying a regression procedure.

Artificial Intelligence approach. *ANN*, *GA*, Bayesian networks (*BN*) and Fuzzy Logic are typical approaches used in *Ra* prediction. In Azouzi and Guillot (1997) the *ANN*, statistical tools, and sensor fusion techniques were combined to estimate *Ra* and dimensional deviations during turning process (on-line). The *DoE* considered the following factors: feed, speed, depth of cut, coolant, tool wear, and material properties. A surface recognition system for *Ra* prediction was proposed by Tsai et al. (1999), where an *ANN* model was built by using the spindle speed, feed rate, depth of cut and the vibration average per revolution. Also, in Benardos and Vosniakos (2003) it was presented an *ANN* model able to recognize *Ra* in a face-milling process.

In Khorasani and Soleymani-Yazdi (2017), a general dynamic *Ra* monitoring system in milling operations was developed. Based on the experimentally acquired data, the milling process was simulated. Cutting parameters (i.e., cutting speed, feed rate, and depth of cut), material type, coolant fluid, X and Z components of milling machine vibrations, and white noise was used. First neural milling process model was developed with 6 inputs, 12 neurons in hidden layers and 3 outputs. The neural milling process model was used to estimate two components of vibrations and *Ra*. Later, a neural simulator model was developed with 7 inputs, 14 neurons in hidden layers and 1 output, and it was used to predict *Ra*. The average correlation factor was computed for training/testing steps: 99% and 99.7% respectively.

Wu and Lei. (2019) proposed an *ANN* model to predict the *Ra* of *S45C* steel in milling process. The input features to the *ANN* model were feed per tooth, cutting depth, clamping torque of vise, the removed volume accumulation, and the extracted features from the vibration signals (*RMS*, kurtosis, skewness, and the envelope signals on the three-axis).

2.2.3 Cutting Tool Wear Condition (CTWC) Modeling

Monitoring the *CTWC* is very important in all metal cutting processes. There are two monitoring systems: direct and indirect. Direct monitoring systems are not easily implemented because they need ingenious measurement methods. Therefore, indirect measurements are required for the estimation of the *CTWC*, and different sensors must be used for monitoring and diagnosing the *CTWC*. There are important contributions to assess the *CTWC* by applying indirect methods: *ANN*, *BN*, and hidden Markov Models (*HMM*).

Artificial Neural Networks. *ANN* are well known as pattern recognition engines and robust classifiers. *ANN* has been used successfully to combine inputs from multiple sensors for *CTWC*. In Haber and Alique (2003) was developed an intelligent supervisory system for *CTWC* prediction using a model-based approach. First, an *ANN* model was trained considering the cutting force, the feed rate, and the radial depth of the cut. Secondly, the residual error obtained from the measure and predicted force was compared with an adaptive threshold to estimate the *CTWC*. This condition was classified as new, half-worn, or worn cutting tool.

In Saglam and Unuvar (2003), the authors worked with multilayered *ANN* for the monitoring and diagnosis of the *CTWC* and *Ra*. The obtained success rates were of 77% for tool wear and 80% for *Ra*.

In Javed et al. (2012) an improved-extreme learning machine (*Imp- ELM*) algorithm was developed for predicting the *CTWC* of a high-speed *CNC* machine. The *ELM* approach was evaluated using the force-based approach to build a prediction model. The implemented *ANN* architecture in the *Imp-ELM* approach was defined by four inputs, 15 to 25 neurons in the hidden layer, and one neuron in the output layer. The experimentation was made in a high-speed *CNC* milling machine, the work-piece material was Inconel-718, and three cutting tools of tungsten carbide with 6 mm ball-nose/3-flutes were used in the milling operation. The result was not good enough when the unknown data was presented.

In Escajeda-Ochoa et al. (2019) a new proposal for monitoring the *CTWC* was developed by using *ANN* with autoencoders. The recorded signals (vibration, cutting forces, and autoencoders during the machining of aluminum workpieces were pre-processed to obtain the *MFCC*. The raw signals were recorded during the machining process by using four *CTWC*: *New, Half-New, Half-Worn and Worn*. The processed features of the process signals were used to build a stacked space autoencoders neural network. During the training of the *ANN*, important parameters that define the sparsity, regularization, and impact of the sparsity regularizer were adjusted to achieve optimal training avoiding the under-fitting and over-fitting. The proposed methodology allows identification of *TWC* with an accuracy of 99.63%.

In Guo et al. (2022), a novel Multi-scale Pyramid Attention Network (*MPAN*) was proposed which can not only accurately monitor tool wear, but also access the interpretability from aspect of model design and feature extraction. The *MPAN* was constructed according to the periodic characteristic of the two sensory signals, cutting force and vibration, which improves the ability to characterize multiple

frequency bands and maintains the periodic meaning of the features to a large extent. The *MPAN* consists of following components: dual-frequency decomposition, dual-frequency attention, pyramid attention and multi-layer perceptron. In two *HSM* experiments with multiple milling conditions, the mean absolute prediction error of *MPAN* was 3.8% and 11.2% respectively, which shows robustness against tool types, milling parameters, and time-varying signals.

Bayesian Networks. *BN*, also known as belief networks, belong to the family of probabilistic graphical models. *BN* combine principles from graph theory, probability theory, computer science, and statistics. In Dey and Stori (2004), a monitoring and diagnosis approach based on a *BN* was presented. This approach integrates multiple process metrics from sensor sources in sequential machining operations to identify the causes of process variations. It provides a probabilistic confidence level of the diagnosis. The *BN* was trained with a set of 16 experiments, and the performance was evaluated with 18 new experiments. The *BN* diagnosed the correct state with a 60% confidence level in 16 of 18 cases.

Abellán et al. (2006) proposed an indirect monitoring multi-sensor system for both *Ra* and *CTWC* using *BN* techniques. In Allen (2006) a *CTWC* monitoring system for milling operation was developed based on the current signal of the spindle motor as the fault indicator signal. Wavelet time-frequency analysis method was employed for the signal processing step. Gaussian process regression, support vector regression, Bayesian rigid regression, nearest neighbor regression, and decision tree methods were implemented to learn a model between tool wear behavior and signal indicators. The decision tree method and Bayesian rigid regression algorithms showed accurate results with accuracy of up to 91.6% for *CTWC* with a promising ability to tolerate and work under changing operating conditions.

Hidden Markov Models. These models are referred to as Markov sources or probabilistic functions of Markov chains. *HMM*s have proved to be indispensable for a wide range of applications in such areas as signal processing, bioinformatics, image processing, linguistics, and others, which deal with sequences or mixtures of components.

In Atlas et al. (2000), the authors used *HMM*s for the evaluation of tool wear in milling processes. The feature extraction from vibrations signals was the root mean squared, the energy, and derivative. Two *CTWC* were defined: worn and no-worn condition. The reported success was around 93%.

Fish et al. (2003) proposed a *HMM*-based classifier that models the dynamics of cutter wear and the dynamics within a cutting pass in a milling application. Each *HMM*s state corresponds to a range of wear, allowing to produce a quantized estimate of cutter wear.

A proposal to exploit speech recognition frameworks in monitoring systems of the *CTWC* was presented in Vallejo-Guevara et al. (2005). Also, Vallejo et al. (2006) introduced a new approach for online monitoring the *CTWC* in face milling. The proposal was based on a continuous *HMM*s classifier, and the feature vectors were computed from the vibration signals between the cutting tool and the workpiece. The feature vectors consisted of the mel frequency cepstrum coefficients (*MFCC*); the success to recognize the *CTWC* was 99.86% and 84.55%, for the training/testing dataset.

The main contributions of this research are:

- The models that predict *Ra* and *CTWC* consider five aluminum alloys widely used in the aeronautical and automotive industries. The models reported in the consulted research usually use a single material.
- In the works reported for milling operations, the *DoE* only considers straight paths or face milling. This work includes a new parameter that allows defining concave, convex, and straight paths, which are common in peripheral milling operations of molds and a great diversity of parts.

- Regarding the *CTWC*, the research only reports two tool wear conditions (new, worn) for a specific tool geometry. The models that predict the *CTWC* consider four states of tool wear and are evaluated for five different diameters of end millings.
- Finally, the intelligent control system allows predicting the cutting parameters (off-line) to obtain the desired *Ra*, considering the different materials and tools. In *HSM*, it is difficult to have a database with recommendations for specific cutting parameters. Usually, they are determined by trial and error.

2.3 Experimental Set Up

Experiments were carried out in an industrial *HSM* center the HS-1000 Kondia, featuring a 25 KW drive motor, 3 axis, 24,000 rpm maximum spindle speed and a Siemens Open Sinumerik 840D controller, and several sensors were installed in the *CNC* machine as shown in Figs. 2.1- 2.2.

Two data acquisition boards acquired the signals with sample rates of 40,000 and 1,000,000, respectively. A milling process was carried with different aluminum alloys, several cutting tools 25^o helix angle, and 2-flute, Sandvik Coromant and several geometries (concave, convex or straight path), Fig. 2.2. Table 2.1 defines the variables and their description used in the *DoE*.

Table 2.1: Definition of variables

Variable	Description	Variable	Description
V_f	Feed rate	HB	Brinell Hardness
n	Spindle speed	Ra	Surface roughness
ap	Axial depth of cut	Ra^p	Predicted Ra
ae	Radial dept of cut	Ra^d	Desired Ra
Curv	Curvature of the geometry	V_B	Flank wear in cutting tools
D_{tool}	Cutting tool diameter	C_c	Cutting conditions: n, V_f, ap, ae
f_z	Feed per tooth	P_c	Cutting parameters (cutting tool, workpiece hardness, etc)
F_y	y-axis workpiece cutting force	P_G	Geometric parameters (cutting tool, path of the cutting process).

2.4 Experimental Database

Most research work related to *Ra* and *CTWC* only considers a specific combination of tool and workpiece material. However, several authors have pointed out the importance of building databases with different materials, cutting tools, and a broader domain in the machining process. In the mold/die industry, the peripheral milling process is a crucial cutting process; the geometric path can be defined as a simple straight line, concave, or convex curvature. Furthermore, in the aeronautic and automotive industries, different aluminum alloys are used according to multiple requirements and applications of the workpieces.

The several groups of factors that affect *Ra* must be considered. The first group is the cutting conditions (cutting speed, feed rate, axial depth of cut, etc.); the second group is the geometry of the cutting

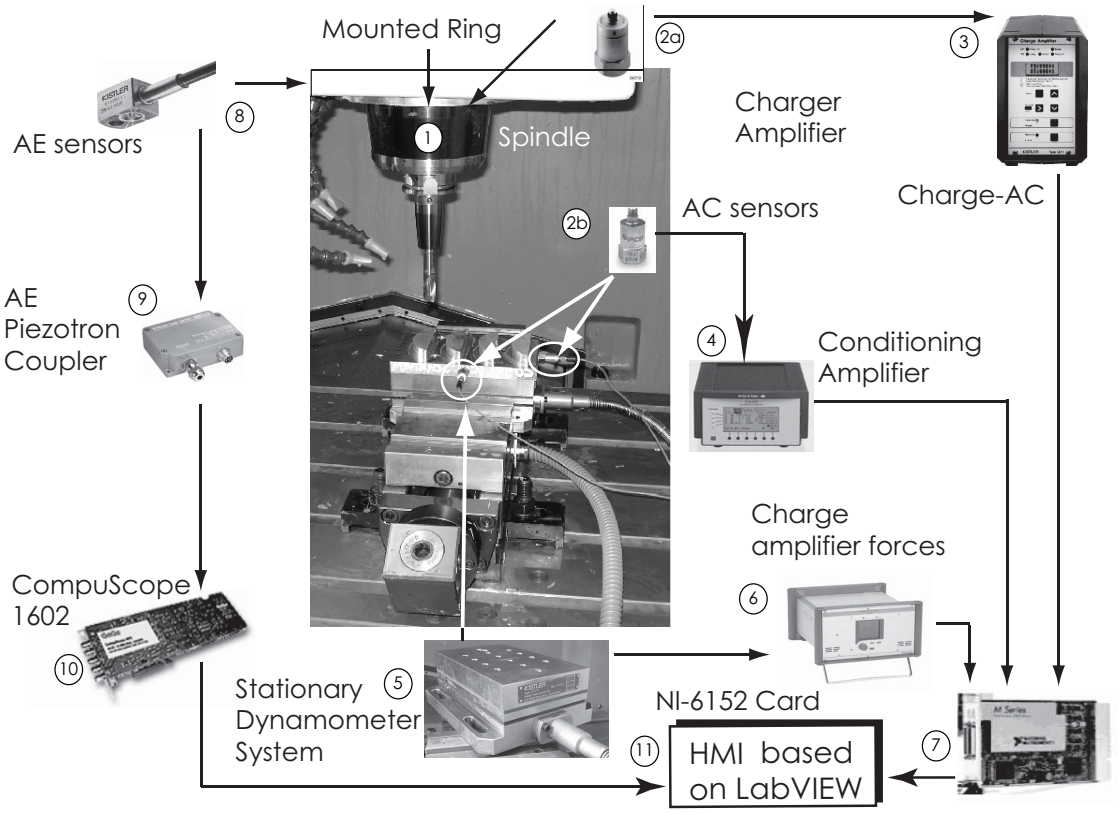


Fig. 2.1: Data acquisition system. The monitoring system is integrated by: (1) Supporting ring, (2a) Brüel & Kjaer accelerometers (charge sensitivity: 98 $+/-$ 2 pC/g, resonant frequency: 16 KHz & 42 KHz), (2b) PCB piezotronic accelerometers (x-axis & y-axis, sensitivity: 10 mV/g; range frequency: 0.3520,000 Hz), (3) Charger amplifiers, (4) Conditioning amplifier, (5) Kistler 3-component forces Dynamometer (force range: -7.5 pC/N in x-axis and y-axis and -3.5 pC/N in z-axis, natural frequency: 3-5 KHz), (6) Kistler charger amplifier, (7) *NI* data acquisition card, (8) Kistler Piezotron *AE* (sensitivity: 700 V/m/s, freq. range: 50-400 KHz), (9) *AE* Piezotron coupler. (10) CompuScope card, and finally, (11) a *HMI* based on *LabView* for real time control and monitoring operations was developed.

tool (tool, diameter, helix angle, edge radius, number of flutes, etc.); the third group is the work-piece material (hardness, ductility, etc.) and path of the peripheral milling process (concave, convex or straight path); the last corresponds to the uncertainty of the process due to the variations in machine vibrations and repeatability, work-holding devices, and other factors. Other variables are less critical or can be controlled (coolant, thermal conditions, humidity, and so on).

A process characterization (screening) design was required in the first experimental stage to compute the critical variables that affect the *Ra*. The *DoE* was calculated by considering eight factors, two levels, 1/8 fraction, 32 runs, zero center points, and four replicates. The selected factors and levels are shown in Table 2.2.

The next step is to apply an analysis of variance for analyzing the significance of factors and/or model terms that address the multiplicity of the tests with the experimental results. The final factors that were

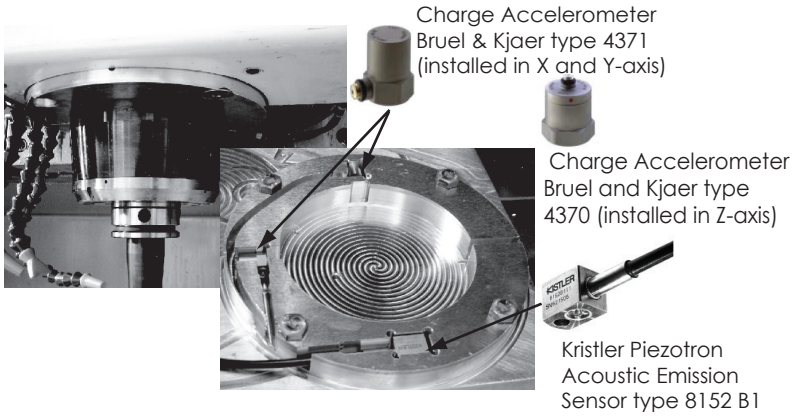


Fig. 2.2: Accelerometers and *AE* sensors installed on a ring fixed to the spindle of the *CNC* machining center.

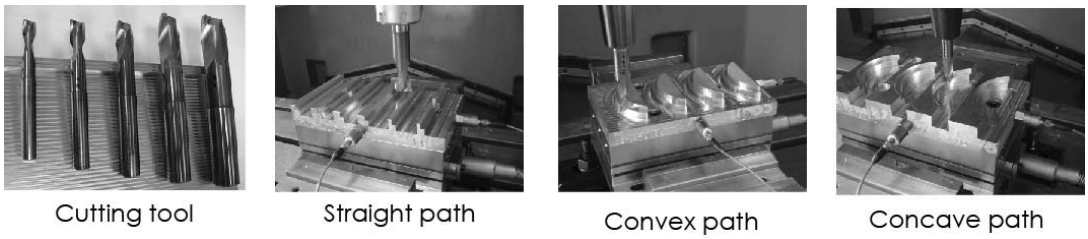


Fig. 2.3: Cutting tools (Sandvik Coromant 8,10,12,14,16 mm) and several geometries (concave, convex, and straight path).

Table 2.2: Factors and levels are defined for the screening design.

Factors	Low level	High level	Units
f_z	0.04	0.13	mm/rev/tooth
V_c	500	850	m/min
ap	5	10	mm
D_{tool}	12	16	mm
ae	1	5	mm
HB	65	145	HB
R	20	40	mm
I/C	Convex	Concave	dimensionless

selected from this analysis are: f_z, D_{tool}, ae, HB, R, and I/C. The last two factors are fused into a new factor called curvature (*Curv*). This factor is computed as the inverse of the radius of the work-piece geometry, and the curvature type is defined by its sign: positive for the convex path and negative for the concave path.

Then, *RSM* is applied to define the runs of the experiments. *RSM* is recommended for modeling and analyzing models with several factors that significantly influence a response. It is based on the central composite design for the design of experiment matrices (Allen, 2006; Montgomery, 2001). Central composite design consists of a 2^{k-1} fractional factorial of resolution with n_F runs and n_c center runs. The defined parameters to apply *RSM* were the following: rotatable central composite design, with 16

points (2^{k-1} = 16, k = number of factors) on cube, 10 points outside of cube, six central points, $\alpha = 2$ as the radius of the sphere, 32 runs, and four replicates. Table 2.3 shows the factors and levels for the experiments.

Table 2.3: Factors and levels of the experimentation with *RSM*

levels	f_z $\frac{mm}{rev}$	D_{tool} mm	ae mm	HB HBN	$Curvature$ mm^{-1}
-2	0.025	8	1	71	-0.05
-1	0.05	10	2	93	-0.025
0	0.075	12	3	110	0
1	0.1	16	4	136	0.025
2	0.13	20	5	157	0.05

The selected aluminum alloys were: 5083-H111, 6082-T6, 2024-T3, 7022-T6, and 7075-T6. The cutting tool diameters (Sandvik Coromant) were: 8, 10, 12, 16, and 20 mm. The workpiece size was 100 mm x 170 mm x 25 mm, and it was designed to allow the machining of four experiments, as seen in Fig. 2.4. The experimentation considered the following steps:

1. Run the 32 experiments with a sharp cutting tool.
2. Record the process variables and measure the *Ra* for the four replicates.
3. Machine harder aluminum alloys until the cutting tool reaches specific flank wear. The following *CTWC* was defined as the half-new cutting tool.
4. Run another set of experiments with the half-new *CTWC*.
5. Repeat steps 1, 2, and 3 by using half-worn and worn *CTWC*. The final *CTWC* was defined until *Ra* presents considerable damage or the cutting tool reaches the maximum tool-life criterion [ISO 8688-2:1989, 1989].

During the experimentation with the four different *CTWC*, the following process variables were recorded:

- Acceleration signals in *x* and *y-axis* directly on the workpiece ($Acc_{x,wp}$, $Acc_{y,wp}$).
- Acceleration signals in *x*, *y*, and *z-axis* directly on the *CNC* spindle ($Acc_{x,sp}$, $Acc_{y,sp}$, $Acc_{z,sp}$)
- Force signals in *x*, *y*, and *z-axis*.
- *AE* signals in the spindle and table of the *CNC* machine.
- The *Ra* was measured in a specific area of the machining surface at the end of each test.

The full database is available online with all the measurements and the corresponding combinations of cutting parameters, as well as the *Ra* values for the different *CTWC*. In addition, authors make the experimental database available [request it via : avallejo (at) tec (dot) mx].

2.4.1 Surface Roughness Database

The quality of the machined surface is characterized by the accuracy of its manufacture concerning the dimensions specified by the designer. Therefore, *Ra* is important because it defines if a workpiece is accepted or rejected after the machining process. Several parameters describe the *Ra*: *Rz*, *Rt*, *Rq*, and

Ra. The most crucial parameter is *Ra*; it is defined as the arithmetical mean of the absolute ordinate values within a sampling length, $Ra = (1/l)\int_0^l |z(x)|dx$, *Ra* was measured with a portable Surfcom type 130A. The sampling length and evaluation length were 0.8 and 4 mm. An area was specified for the measurement of *Ra*, Fig. 2.4 (red circle). The procedure used to measure the *Ra* is depicted in Fig. 2.5.

Fig. 2.4: Identification of the area for measuring the *Ra* in the specimens

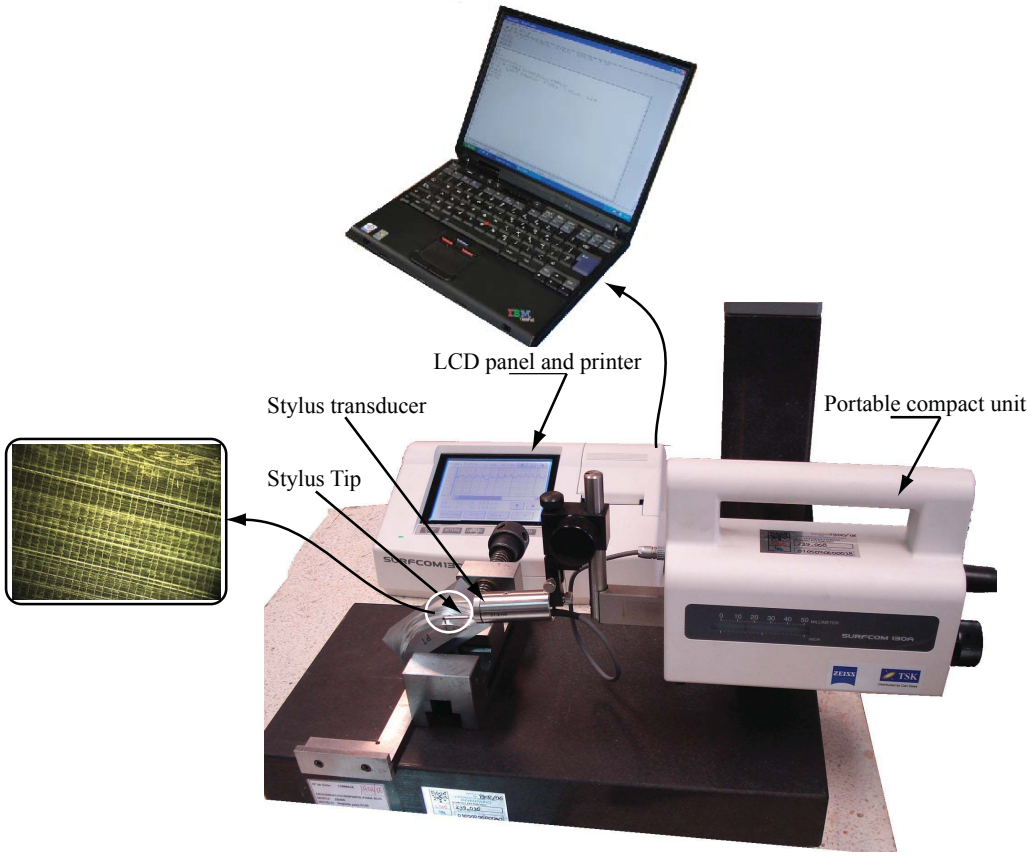


Fig. 2.5: (a) Measurement of *Ra* with the stylus tip of the surf com 130 A; (b) Register and printer of the information; (c) PC to store the information; (d) Image of the machined surface.

Additional to the computed *Ra*, the different signals were processed by using the *MFCC* technique Vallejo and Morales-Menendez (2010) to obtain seven *Cepstrum Coefficients*, and *ANN* models were used to on-line predict *Ra*, during the machining process, Vallejo-Guevara et al. (2009c).

2.4.2 Cutting Tool Wear Condition Database

During the machining process of the *DoE*, the machining time, the volume of the removal metal, and the number of cycles of the cutting tools were computed. Also, the flank wear evolution was measured during the experimentation. Figure 2.6 shows the equipment used to measure the flank wear of the cutting tool. The tool-life criterion was defined according to the international standard ISO-8688-2, *Tool life testing in milling. Part 2: End milling*. The tool life criterion was considered based on the flank wear of the tool edges. The uniform flank wear (VB_{avg}) represents the average value of the two cutting tool edges, and the maximum flank wear (VB_{max}) corresponds to the higher value found in the cutting edges. The recommended values of the flank wear are the following: (1) Uniform average wear of 0.3 mm over all the edges of the cutting tool, and (2) Maximum wear value of 0.5 mm on any individual edge or tooth of the cutting tool. The flank wear values considered to classify the four *CTWC* are shown in Table 2.4.

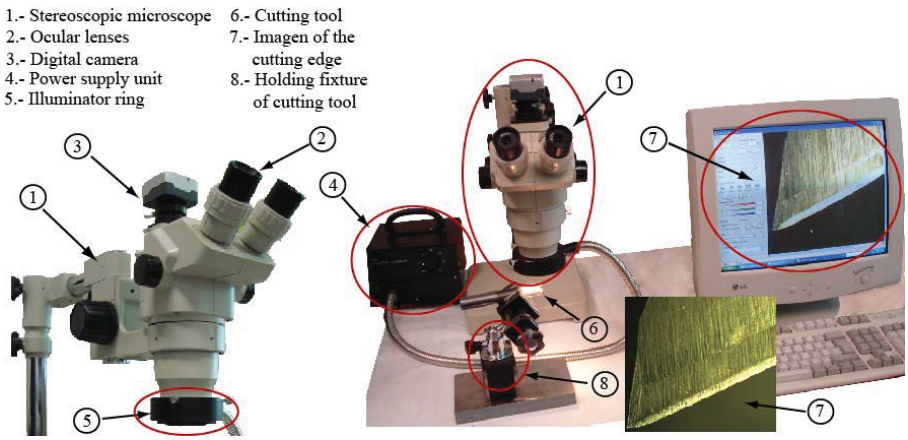


Fig. 2.6: Measurement of *Ra* with the stylus tip of the surf com 130 A. The information was recorded and printed in the LCD panel and printer, and stored in a PC.

Table 2.4: *CTWC* and the flank wear observed during the experimentation

CTWC	Uniform flan wear (mm)
New	$0 \leq V_B < 0.07$
Half-new	$0.07 \leq V_B < 0.1$
Half-worn	$0.1 \leq V_B < 0.18$
Worn	$0.18 \leq V_B < 0.45$

Two approaches were used to monitor and diagnose the *CTWC*: *HMM* and *ANN*, both approaches used the following information to build the models.

1. The five cutting parameters: f_z, D_{tool}, ae, HB, $Curv$.
2. The seven *Cepstrum Coefficients* of the different process signals (acceleration in the workpiece, acceleration in the spindle, the cutting forces, and *AE* in the spindle and table).
3. The *CTWC*s: *New, Half-new, Half-worn,* and *Worn.*

Considering all the experiments carried out for the four states of the tool, there are 441 data for each process signal. Therefore, the input data must be grouped in a 441×12 matrix, and the output data must be grouped in a 441×4 matrix.

2.5 Intelligent Control System

Several industrial projects had been developed and end with the need of a *control system*, Fig. 2.7, Vallejo-Guevara et al. (2005), Vallejo-Guevara et al. (2007), Vallejo-Guevara et al. (2008). This control system integrates four main modules: (1) data acquisition, (2) *CTWC*, (3) *Ra* monitoring and (4) planning module.

2.5.1 Data Acquisition Module

Based on the data acquisition system, Fig. 2.1, standard filtering was applied to the process variable signals. Some signals were pre-processed by the *MFCC*, widely used in speech recognition systems, Wong and Sridharan (2001); to find particular features of the data (**Appendix 1** shows the computing procedure.)

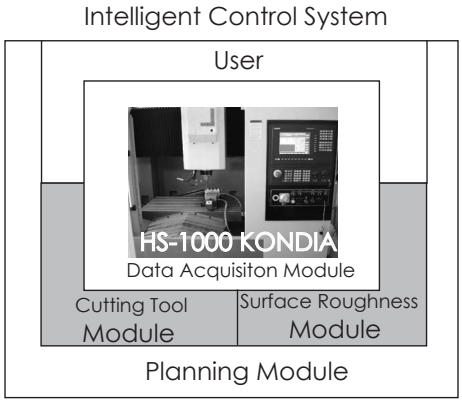


Fig. 2.7: Intelligent Control System with four modules: data acquisition, *CTWC*, *Ra* and planning module

2.5.2 Cutting tool Module

The *CTWC* has a direct impact on the final dimensions of the product, *Ra* and surface integrity. However, direct monitoring is not easily implemented due to nonstandard measuring methods.

An indirect monitoring approach based on measurement of different process signals was developed Vallejo-Guevara et al. (2007), Vallejo-Guevara et al. (2008), and Vallejo-Guevara et al. (2005). The process signals were characterized by *MFCC* and associated with the *CTWC*. The *CTWC* were defined as: New ($0 \leq V_B < 0.07$ *mm*), Half-new ($0.07 \leq V_B < 0.1$ *mm*), Half-worn ($0.1 \leq V_B < 0.18$ *mm*), and Worn ($0.18 \leq V_B < 0.45$ *mm*), where V_B is the flank wear according to ISO-8688-2 norm. A *HMM* framework was developed to identify based only on the *MFCC* of the process signals. Additionally, an *ANN* model was implemented to classify the *CTWC*.

2.5.3 Surface Roughness Module

It is also possible to predict *Ra* during the machining process by applying an *ANN* model. The *ANN* model was built based on cutting parameters (f_z, D_{Tool}, *ae*, *HB*, *Curv*) and on-line measurement of process variables. The *MFCC* were computed for different process signals, and the *ANN* model was defined as $Ra = ANN(f_z, D_{Tool}, ae, HB, Curv, MFCC)$. Each model was developed for a *CTWC*, verifying that the residuals followed a normal distribution statistically validated. An estimator based on multi-sensor and data fusion provides an improved and robust estimation, Vallejo-Guevara et al. (2009b).

2.5.4 Planning Module

A *CNC* machining center must consider several intelligent areas, such as, *CTWC* monitoring, operation & machine tool modeling and adaptive control, Monostori (2000). The planning module has two main tasks: computation of the optimal cutting parameters that minimizes *Ra* (pre-process and in-process), and computation of the machining policy that minimizes the production cost.

Cutting Parameters (off-line optimization). One of the key tasks of the planning module is the computation of the optimal cutting parameters before the cutting operation (*off-line optimization*). Given a set of variables provided by the operator (C_c, P_c, P_G, R_a^d) the surface roughness (R_a^P) is estimated, and the cutting parameters are optimized with a *GA*. Left picture of Fig. 2.8 shows the detailed procedure.

Cutting Parameters (on-line optimization). The second key task of the planning module is the computation of the optimal cutting parameters during the machining process (on-line optimization). Considering some process variables and the on-line estimated V_B, the actual *Ra* (Ra^p) is estimated. First, the difference between the Ra^p and the desired *Ra* (Ra^d) is computed; based on this error and the previous feed per tooth, the new *Ra* is re-computed. Finally, the Ra^p is re-estimated on-line, based on current process, see right picture of Fig. 2.8.

Machining Policy. The third essential task of the planning module is the optimization of the machining policy, based on a minimization of the production costs. The policy, which generates guidelines for

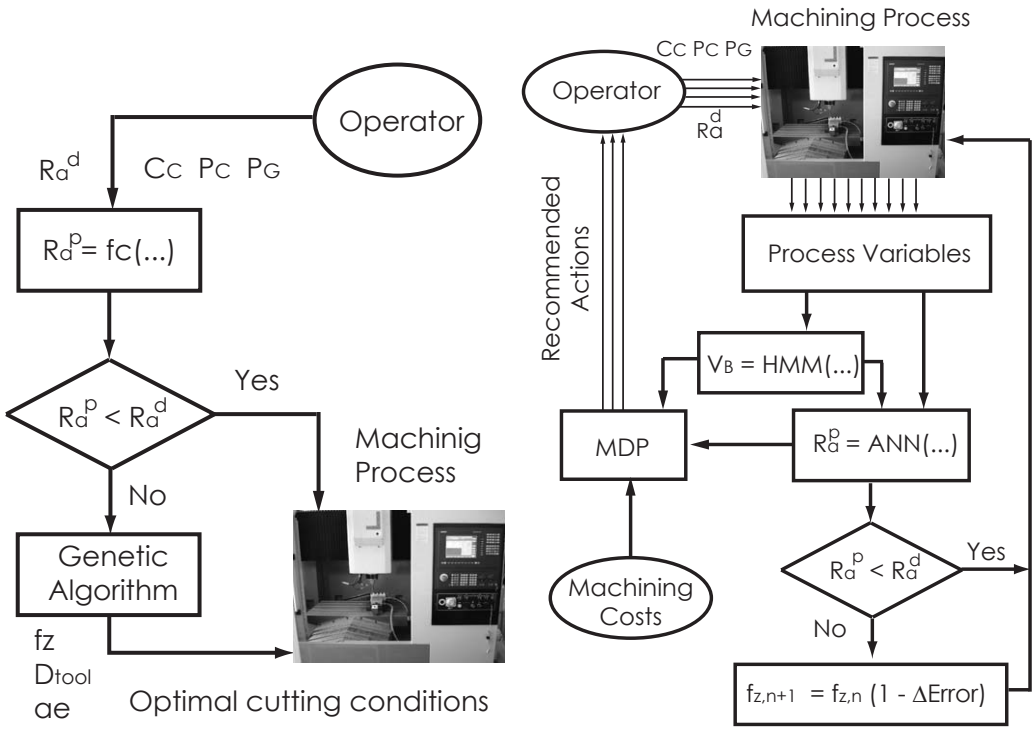


Fig. 2.8: Off-line optimization (left picture) and on-line optimization (right picture) of cutting parameters and machining policy

the operator is limited to the available universe of variables of this problem (different Aluminum alloys, cutting tool diameters, and *CTWC*s).

A methodology based on the *MDP*, Feldman and Valdez-Flores (2004) was implemented. The key characteristic of a Markov model is a probability law in which the future behavior of the system is independent of the past behavior, given its current condition. A formal description of a *MDP* is as follows: $S = \{s_1, s_2, s_3, s_4, s_5\}$ is a finite set of states of the system. The possible states of the *CTWC* are: s_1, new; s_2, half-new; s_3, half-worn; s_4, worn, and s_5, tool fracture, and $A = \{a_1, a_2, a_3\}$ is a finite set of actions that the operator can take. The possible actions are described below.

* a_1, no action. An aggressive condition, because the operator uses the cutting tool until to reach the V_B maximum.
* a_2, change the cutting tool. A conservative condition, which implies to change the cutting tool when the *CTWC* module predicts the worn condition.
* a_3, stop the machine and inspect the cutting tool.

P: $S \times A$ is the state transition probability distribution function. For each action and state of the system, there is a probabilistic distribution over the states that can be reached after the actions. The function $P(s|s',a)$ is defined as the probability of reaching state s starting in-state s' and given action a. As shown in Fig. 2.9, the transition matrices were computed by considering the evolution of the cutting tool's longevity.

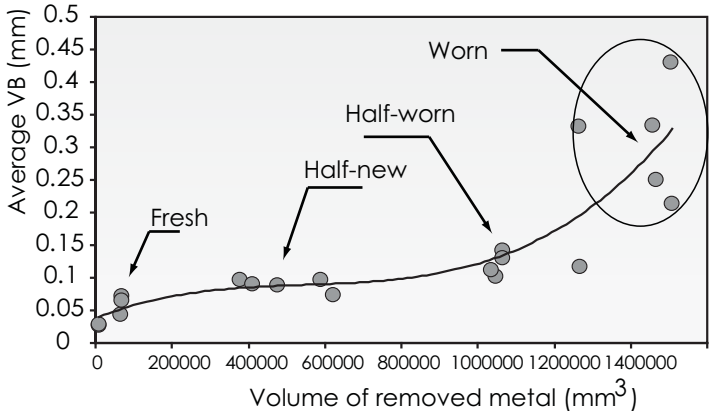


Fig. 2.9: Four states of the *CTWC* can be identified. The measured V_B is a function of the removal metal.

The tool fracture was included to simulate a random failure of the cutting tool, which can happen during the machining process. The instantaneous cost function must be defined for each action. Cost functions are computed by considering: a) the decision cost for a right or wrong action, b) operator costs, energy cost, and the operator labor, and c) the cost of the cutting tool. The cost function was defined for all the *CTWC*s and for each action.

$R : S \times A$ is a reward function for executing an action in the state s, assigning a real number for each action in each state of the system. β defines a vector that maps the state space into the action space, that is, an action function, which assigns an action to each state. These are evaluated by the *MDP* algorithm to compute the optimal policy. A stationary policy (π) is a policy that an action function can define. The stationary policy is defined by the function (β) taking action $a(i)$ at time n, if $S_0 = i$, independent of previous states, actions and time-steps. The set of all (decisions) policies is denoted by v. The *Expected Discount Cumulative Cost* will be used to compute the optimal minimum cost, see details at Vallejo-Guevara et al. (2009a).

2.6 Results

A novel mathematical framework for fault diagnosis (i.e. cutting tool wear condition, and surface roughness) is one the most important results. It is proposed as a key support of intelligent control under uncertainty.

2.6.1 Surface Roughness Modeling

The *Ra* can be on-line predicted, two different *ANN* models were proposed for *Ra* prediction during the machining process, Fig. 2.10.

Several tests were defined to select the best *ANN* model to estimate the *Ra* as a function of the cutting parameters and some process variables represented by the *MFCC* of the corresponding signal. Among

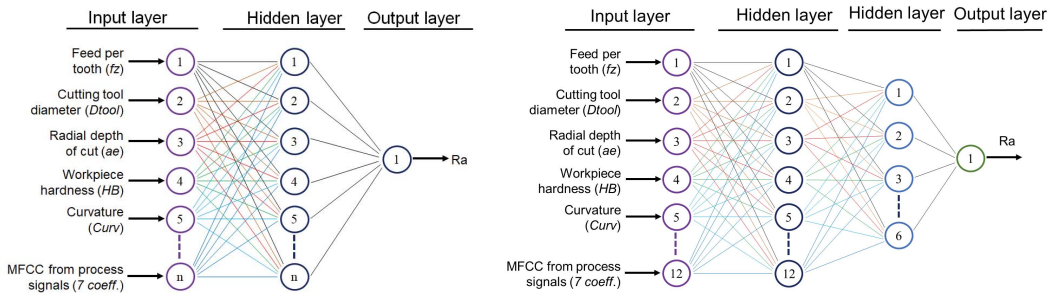


Fig. 2.10: Proposal architectures for the *ANN* models with the input/output variables predicted *Ra*.

the various *ANN* models, feed-forward architecture was selected, and the training algorithm was the *Levenberg-Marquardt*. Two different structures were defined to build the *ANN* model: $12 \times 12 \times 1$ (one hidden layer), and $12 \times 12 \times 6 \times 1$ (two hidden layers). The selected process signals were the following: acceleration in the $x-axis$ of the workpiece, acceleration in the $y-axis$ of the workpiece, acceleration in the $y-axis$ in the spindle, force in the $x-axis$, force in the $y-axis$, and the *AE* in the spindle. Four different models were computed in agreement with the *CTWC*: New, half-new, half-worn, and worn cutting tool condition. The procedure to compute the *ANN* model by considering the cutting parameters and *MFCC* of each signal was as follows.

1. From the database, file must be selected with the input/output variables for a specific process signal.
2. Separate the input/output variables in two files (*MatLab* format). The input variables are stored in a 12×110 matrix and the output variable was stored in a 1×110. The 110 experiments were stored randomly, together with the corresponding *Ra* value.
3. The input/output information of variables was divided into two data sets for the training (70%) and testing (30%) procedures.
4. The feedforward network was constructed, and it was trained with the *Levenberg-Marquardt* algorithm.
5. The estimation of *Ra* was computed with the trained network, given the training data set.
6. The estimation of *Ra* was also computed for the testing data set.
7. A correlation factor (R^2) was computed between the network response and the corresponding target vector, ($R^2 = 1$ means perfect correlation).
8. The trained network performance was computed with the mean squared error between the estimated *Ra* and the target of the testing data set.
9. The average performance and generalization capacity of the *ANN* models were computed by considering four random combinations of the input/output data.

Table 2.5 presents the average correlation factor for two *ANN* models by using the training data set.

Table 2.6 shows the average correlation factor (R^2) and the *MSE* computed between the estimated and target *Ra* values. The results correspond for the testing data set.

For the new *CTWC*, the *ANN* model with the best performance was obtained by using the *AE* signal and the cutting parameters. Figure 2.11 shows the *Ra* estimated and target values for the testing data. The correlation factor (R^2) was 93.15% for the testing data set. For the half new tool condition, the *ANN* model with the best performance was computed with force in the *y-axis*. The correlation factor

Table 2.5: Average correlation factor (R) for the trained *ANN* models with the training data set.

Cutting Tool Wear Condition	Model	Correlation factor (%) for training data set					
		AccX-workpiece	AccY-workpiece	AccX-spindle	Force-X	Force-Y	AE-Spindle
New	$ANN(12\times12\times1)$	94.4	86.14	89.09	92.31	89.47	91.01
	$ANN(12\times12\times6\times1)$	92.24	95.01	95.14	96.23	94.96	96.07
Half-New	$ANN(12\times12\times1)$	93.7	93.21	94.3	93.74	94.59	93.75
	$ANN(12\times12\times6\times1)$	93.24	95.04	94.72	94.6	93.91	91.67
Half-worn	$ANN(12\times12\times1)$	90.62	91.59	92.37	93.16	93.95	90.21
	$ANN(12\times12\times6\times1)$	93.61	90.89	94.48	94.29	95.02	92.25
Worn	$ANN(12\times12\times1)$	93.38	93.85	95.39	94.13	96.03	95.44
	$ANN(12\times12\times6\times1)$	95.89	94.19	94.35	95.21	96.55	95.19

Table 2.6: Average correlation factor (R) for the trained *ANN* models with the testing data set.

Cutting Tool Wear Condition	Model	Correlation factor (%) and MSE (%) for testing data set											
		AccX-workpiee		AccY-worpiece		AccY-spindle		Force-Y		Force-X		AE-spindle	
		R	MSE	R	MSE	R	MSE	R	MSE	R	MSE	R	MSE
New	$ANN(12\times12\times1)$	86.87	0.46	84.44	0.61	77.92	0.93	83.75	0.69	73.49	1.39	83.48	0.62
	ANN $(12\times12\times6\times1)$	91.68	0.41	90.14	0.3	88.54	0.37	89.74	0.4	85.95	0.57	92.27	0.25
Half-New	$ANN(12\times12\times1)$	82.8	0.52	83.25	0.57	86.72	0.35	86.24	0.45	84.9	0.49	84.6	0.49
	$ANN(12\times12\times6\times1)$	87.15	0.36	87.64	0.37	88.24	0.31	88.5	0.29	89.49	0.3	84.77	0.38
Half-worn	$ANN(12\times12\times1)$	81.32	0.75	73.51	1.14	81.55	1.13	82.19	1.25	87.37	0.64	83.33	1.45
	$ANN(12\times12\times6\times1)$	81.72	0.74	80.75	0.96	84.45	0.75	82.75	0.75	87.83	0.52	84.45	0.62
Worn	$ANN(12\times12\times1)$	80.35	0.39	79.19	0.6	83.94	0.34	86.16	0.34	87.24	0.27	84.86	0.36
	$ANN(12\times12\times6\times1)$	86.61	0.25	84.09	0.38	88.16	0.21	87.98	0.23	88.01	0.23	88.75	0.28

was 89.49% and the *MSE* was 0.30%. For the half-worn tool condition, the *ANN* model with the best performance was obtained with force in the *y-axis*. Finally, for the worn *CTWC*, the *ANN* model with the best performance was computed with the *AE* signal in the spindle.

2.6.2 Cutting Tool Wear Condition Modeling

It is vital to develop an *ANN* model to predict the *CTWC* during the machining process. The *ANN* model must be developed considering the experiments for both new, half-new, half-worn, and worn *CTWC*. The number of experiments was 441. Again, the inputs to the *ANN* model are the five cutting parameters and the seven Cepstrum coefficients of each process signal. The output of the *ANN* model will be the *CTWC*. The data provided in the files to train the *ANN* must be normalized to avoid numerical instability. The data were normalized with a mean zero and a standard deviation equal one. Additionally, the bipolar sigmoidal normalization was employed because the minimum and maximum values are unknown in real-time, and it is given by $\bar{z}_i = (1-e^{-y_i})/(1+e^{-y_i})$.

The defined structure to build the *ANN* model was $12\times12\times6\times4$ (two hidden layers). The selected process signals were the following: acceleration in the *x-y axes* of the workpiece, acceleration in the *y-axis* in the spindle, force in the *x-y axes* and the *AE* in the spindle. The procedure to compute the *ANN* model by considering the cutting parameters and *MFCC* of each signal was the following:

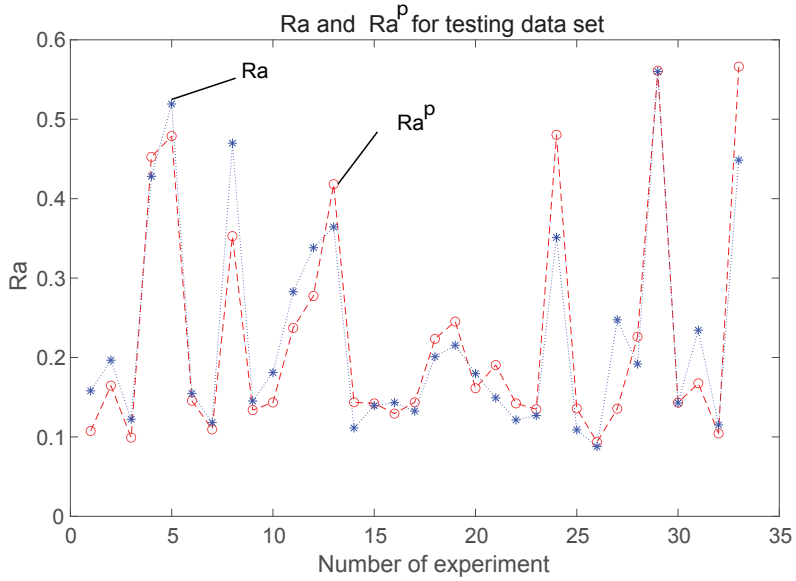


Fig. 2.11: Results of *Ra* estimated and the target values for the *ANN* model and the new *CTWC*.

1. From the database, the file must be selected with the input and output variables for a specific process signal.
2. The input data for the *ANN* model must be normalized with the procedure described previously. The input and output variables must be separated into two files (*MatLab* format). The input variables were stored in a 441×12 matrix, and the output variables were stored in a 441×4 matrix.
3. The input and output variables must be divided into two data sets for the training (70%) and testing (30%) procedures. This step involved selecting 70% of the data for each *New, Half-New, Half-Worn* and *Worn* cutting tool condition. The rest of the data will be used for the testing procedure.
4. The input and output variables were selected randomly.
5. The feedforward network was constructed, and it was trained with the *Levenberg-Marquardt* algorithm.
6. The estimation of V_B was computed with the trained network, given the training data set.
7. The estimation of V_B was also computed for the testing data set.
8. A confusion matrix was computed to show the *ANN* performance.
9. The average performance and generalization capacity of the *ANN* models were computed by considering four random combinations of the input and output data.

Table 2.7 presents the average performance for the *ANN* model by using the training and testing data set. The number of epochs used for convergence is also included.

The best performance of the *ANN* model was obtained by using the *AE* signal. The average performance for the training data set was 100% and for the testing data set was 98.67%. Additionally, the *ANN* model with *AE* process signal converges very quickly.

Figure 2.12 shows the confusion matrix for the training and testing data sets with the *AE* signal. The obtained results show an excellent performance of 98.5% for the testing data set; only two experiments were classified as worn *CTWC*. In reality, these should be classified as half-worn *CTWC*.

Table 2.7: Performance of the *ANN* model for the training and testing data set.

	Acc X-axis workpiece			Acc Y-axis workpiece			Acc Y-axis spindle		
	Performance		Number	Performance		Number	Performance		Number
	Training	Testing	epochs	Training	Testing	epochs	Training	Testing	epochs
	96.1	39.1	1000	98.1	42.9	1000	99.4	60.2	1000
	98.1	38.3	1000	97.4	43.6	1000	100	63.9	1000
	93.2	39.8	1000	97.1	48.1	1000	100	58.6	1000
	95.5	44.4	1000	89	45.9	1000	100	61.7	1000
Average	95.72	40.4		95.4	45.12		99.85	61.1	
	Force X-axis			Force Y-axis			AE in spindle		
	Performance		Number	Performance		Number	Performance		Number
	Training	Testing	epochs	Training	Testing	epochs	Training	Testing	epochs
	99.7	66.9	1000	99.7	66.9	1000	100	100	18
	100	72.2	269	99.7	68.4	1000	100	97.7	21
	99.7	74.4	1000	99.7	70.7	1000	100	98.5	18
	100	71.4	194	99	67.7	1000	100	98.5	19
Average	99.85	71.22		99.52	68.42		100	98.67	

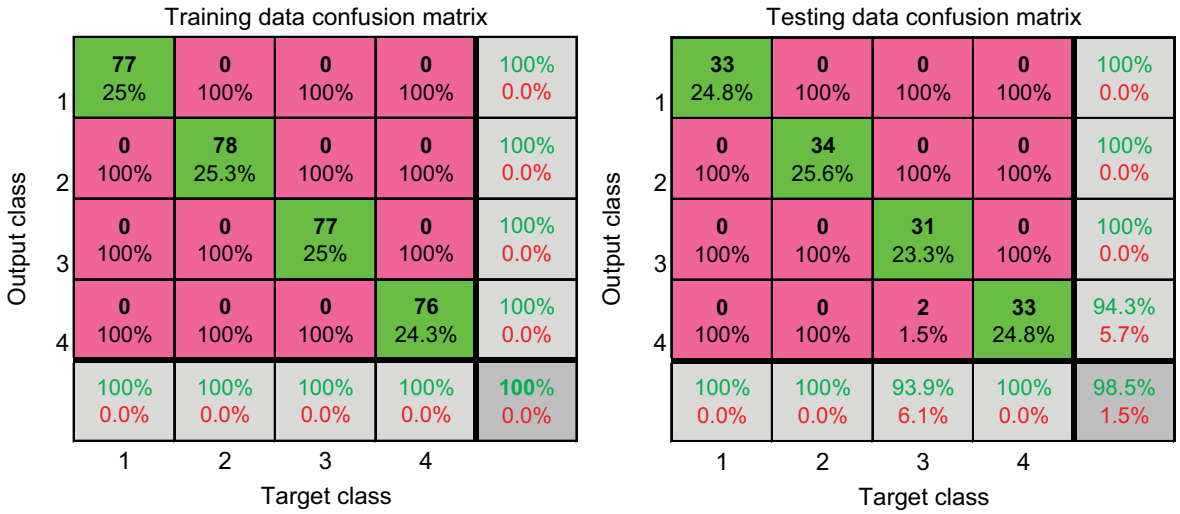


Fig. 2.12: Confusion matrix of the computed *ANN* model with *AE* process signal

Figure 2.13 shows the targets testing data and the response of the *ANN* model to predict the *CTWC*. The results show only the two misclassified experiments.

2.6.3 Off-line Optimization

The optimization step was validated with several tests.

1. An operator defines the cutting conditions, cutting and geometric parameters, and the desired *Ra* value (R_a^d).
2. The planning module computes the R_a^P value, under these conditions.

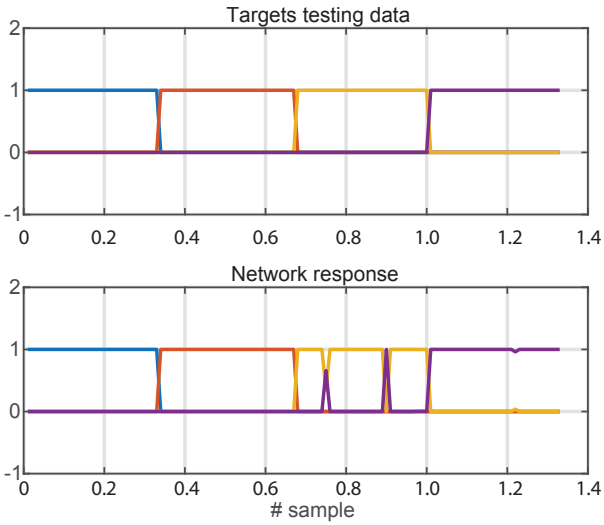


Fig. 2.13: Classification of the testing data set with the *ANN* model using the *AE* process signal

3. If R_a^p is bigger than R_a^d, the *GA* computes the new value of f_z, and the R_a^p value is calculated again, as shown in the left plot of Fig. 2.14.
4. If the new value of R_a^p is still high, the *GA* computes the new values of f_z and D_{tool} that minimize R_a^p, as shown in the center plot of Fig. 2.14. Then, the R_a^p value is calculated with the new cutting parameters.
5. If the R_a^p value continues high, the *GA* executes another iteration to find new f_z and *ae* values, see right plot of Fig. 2.14 . Finally, if the R_a^p value is equal or lower than the R_a^d value, the new cutting parameters are accepted.
6. If the R_a^p value continues high, the work piece material must be replaced, and the procedure must be repeated.

The *GA* was configured by 100 generations, 20 population sizes, 0.8 crossover probabilities, and 0.2 mutation probabilities. The feed per tooth ranged between 0.025 - 0.13 mm/foot, the radial depth of cut ranged between 1 and 5 mm and the cutting tool diameter 8 and 20 mm.

2.6.4 Machining Policy

The *MDP* was validated in the industrial HS-1000 Kondia machining center. The *MDP* can be solved using different algorithms: Police iteration and value iteration. The optimal total cost function was computed based on the defined *MDP*. The optimal policy π can be obtained by an iterating step that defines the actions of the operator that minimize the cost.

Given that *MDP* is a stochastic model defined by a Markov property, the transition matrices, and an initial distribution of the states was simulated several times to illustrate the variability of the results. Figure 2.15 shows two simulations given the (aggressive condition) and (conservative condition) matrices. The left plot of this figure shows a normal evolution of the cutting tool, where the operator does not

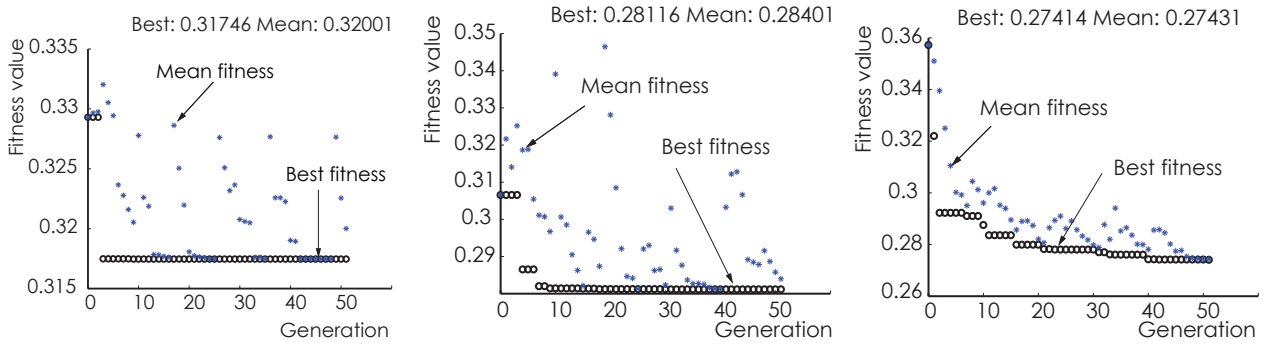


Fig. 2.14: Left plot: *Ra* optimization based on f_z. Center plot: Optimization based on f_z and D_{Tool}. Right plot: Optimization based on f_z and *ae*

take actions and wait for a possible tool fracture when the cutting tool reaches the top worn condition. The right plot of Fig. 2.15 depicts the conservative condition, where the operator decides to change the cutting tool if the worn *CTWC* is detected during the machining process.

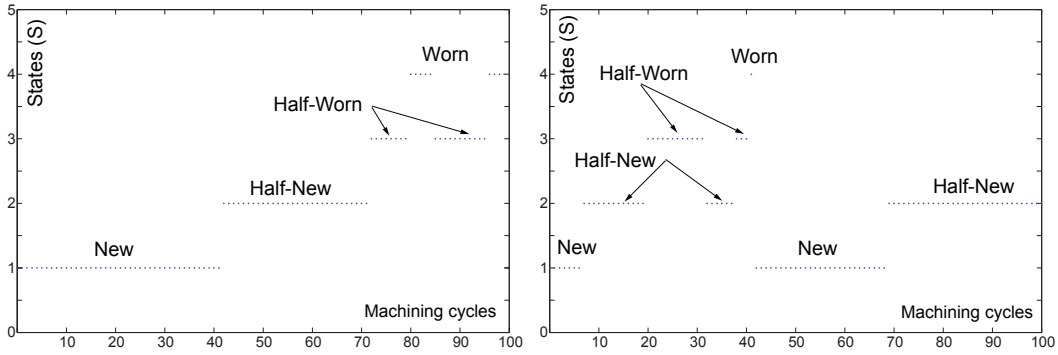


Fig. 2.15: Simulation of the Markov system with the two transition matrices. Left plot presents an aggressive condition for the action a_1. The right plot presents a conservative condition for the action a_2

Figure 2.16 illustrates the results of the variability of the Markov system for 30 evaluations. The *box and whisker* plot shows the comparative costs between the different actions and the optimal policy determined by the *MDP*. These results demonstrate that the the optimal policy presents the lower costs when compared with the aggressive, intermediate, and conservative actions.

2.7 Conclusions

A planning module for *HSM* was designed and incorporated within an intelligent control system. This module is based on the *MDP* framework, yielding novel features in an optimization process. The Intelligent Control System allows the following functions:

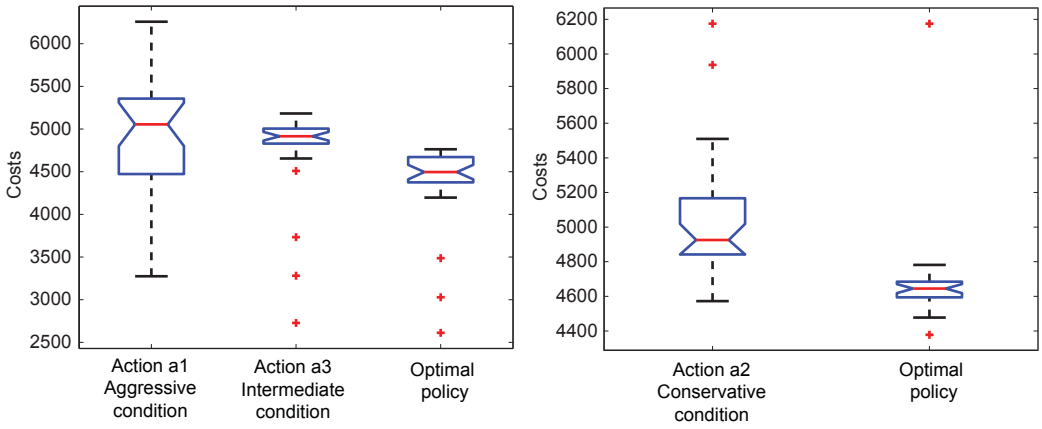


Fig. 2.16: Comparison of costs for the different actions and optimal policy by using the box a whisker plot. Results with Pa_1 (left plot) and Pa_2 (right plot) transition matrices

- In off-line operation, the system estimates the *Ra* and compares it to a desired *Ra*. Using a *GA* determines the optimal cutting parameters to minimize the *Ra* to the desired value.
- In on-line operation, the system uses *ANN* models to estimate the *Ra* and the *CTWC*, which will allow to validate that the desired *Ra* is met. *ANN* use cutting parameters and different process state variables to monitor *Ra* during machining of different aluminum parts and with different cutting tool diameters.
- The *MDP* framework allows modeling decision-making under uncertainty where the actions of the operator are partly under control.

Regarding the limitations of the models and the intelligent control system developed, the following can be mentioned:

- A novel mathematical framework for fault diagnosis (i.e., CTWC and surface roughness) was developed as a crucial element for this module. Although early results are promising, the full integration of an *MDP* framework will require more research into the cross-relationship between key variables.
- Machine tools have multiple degrees of freedom and are mainly related with vibrations and dynamic loading of mechanical structures. It is important to compute performance of the proposal models in different machines tools and estimate deviations with respect to the original *HSM* machining center.

Acknowledgements The authors would like to thank to J.R. Alique (Instituto de Automática Industrial, Spain) for the financial support and technical advice during the experimental program. Authors make the experimental database available

Appendix 1. Mel Frequency Cepstrum Coefficients

The procedure for computing the *MFCC* is given in Fig. 2.17 and can be summarized as follows:

1. A small segment of the signal is selected for applying a Discrete Fourier Transform, to compute the magnitude of the energy spectrum in a logarithm scale.
2. The real frequency scale (f_{Hz}) is mapped to the perceived frequency scale (f_{Mel}) as:

$$f_{Mel} = 2,595 \log\left(1 + \frac{f_{Hz}}{700}\right) \tag{2.1}$$

3. After a triangular band-pass filter is applied for smoothing the scaled spectrum, the *MFCC* are computed using the inverse Discrete Fourier Transform:

$$MFCC_c = \sum_{j=1}^{N_p} y(j) cos\left(\frac{\pi}{N_p}\left(j - \frac{1}{2}\right)c\right) \tag{2.2}$$

where $y(j)$ is the output of the triangular band-pass filter, N_p is the number of band-pass filters, c defines the Cepstrum coefficient number (c = 1,2,...,N_p) , and N_p defines the total number of Cepstrum coefficients.

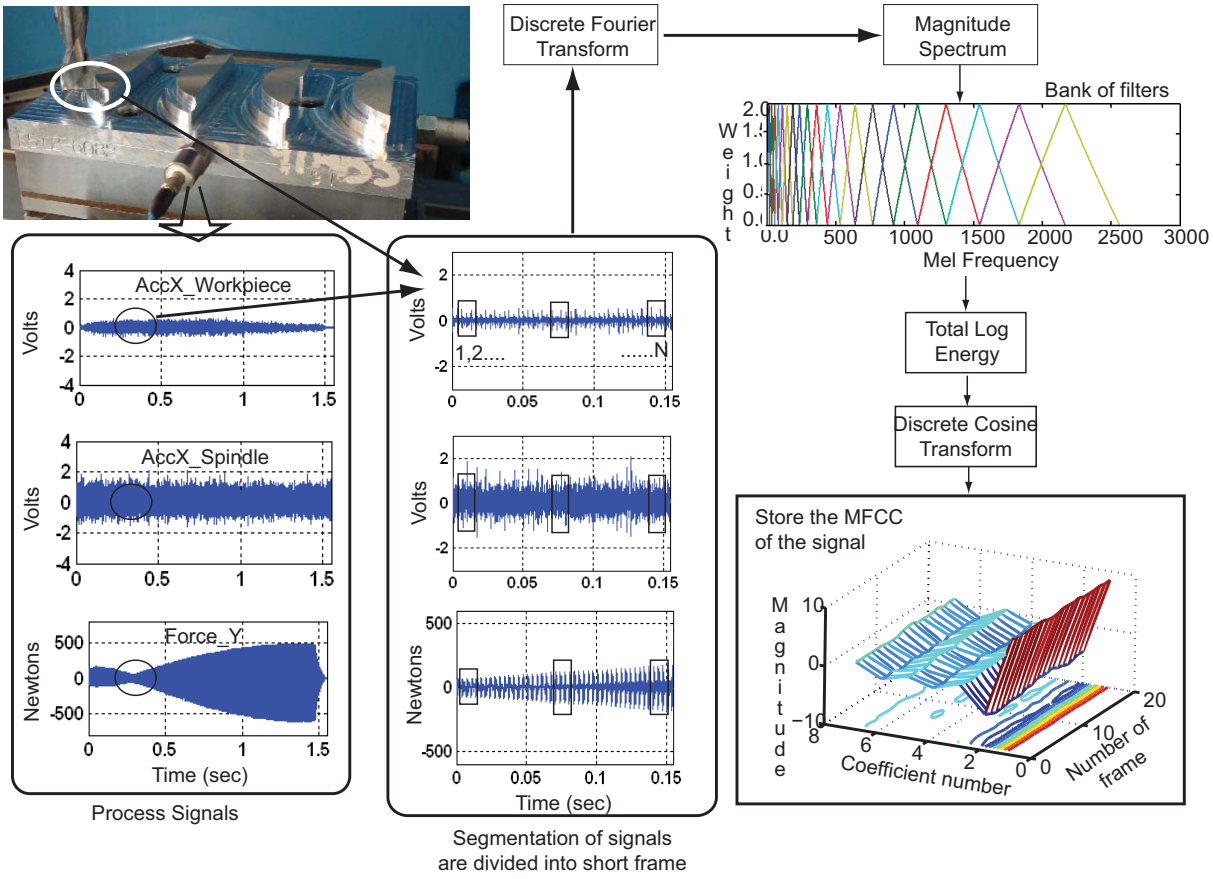


Fig. 2.17: Procedure of the *MFCC* computation

References

Abellán, J., Morales-Menendez, R., Vallejo, A., and Rodriguez-Gonzalez, C. (2006). Surface Roughness and Cutting Tool-Wear Diagnosis based on Bayesian Networks. In 6^{th} *IFAC-SafeProcess*, China.

Abouelatta, O. and Madl, J. (2001). Surface Roughness Prediction based on Cutting Parameters and Tool Vibrations in Turning Operations. *J of Materials Processing Technology*, (118):269–277.

Allen, T. (2006). *Introduction to Engineering Statistics and Six Sigma*. Springer-Verlag, London, 1 edition.

Atlas, L., Ostendorf, M., and Bernard, G. (2000). Hidden Markov Models for Machining Tool-Wear. *IEE*, pages 3887–3890.

Azouzi, R. and Guillot, M. (1997). On-line Prediction of Surface Finish and Dimensional Deviation in Turning using Neural Network based Sensor Fusion . *Int J of Machine Tools and Manufacture*, 37(9):1201–1217.

Balic, J. (2006). Intelligent CAD/CAM Systems for CNC Programming - An Overview. *Advances in Production Eng & Managment*, (1):13–22.

Barber, G., Gu, R., Jiang, Q., and Tung, S. (2001). Surface Roughness Model for Worn Inserts of Face Milling: Part II - An Empirical Model. *Tribology Transactions*, 44(1):142–146.

Benardos, P. and Vosniakos, G. (2003). Predicting Surface Roughness in Machining: A Review. *Int J of Machine Tools and Manufacture*, (1):833–844.

Boothroyd, G. and Knight, W. (2006). *Fundamentals of Machining and Machine Tools*. Taylor and Francis Group, 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL USA, 3 edition.

Carpenter, I. and Maropoulos, P. (2000). Automatic Tool Selection for Milling Operations. Part 1: Cutting Data Generation. In *Proc of the Institution of Mechanical Engineers, Part B: J of Eng Manufacture*, pages 271–282.

Dereli, T., Filiz, I., and Baykasoglu, A. (2001). Optimizing Cutting Parameters in Process Planning of Prismatic Parts by using Genetic Algorithms. *Int. J. Producton Research*, 39(15):3303–3328.

Dey, S. and Stori, J. (2004). A Bayesian Network Approach to Root Cause Diagnosis of Process Variations. *Int J of Machine Tools & Manufacture*, (45):75–91.

Escajeda-Ochoa, L., Ruiz Quinde, I., Chuya Sumba, J., Vallejo-Guevara, A., and Morales-Menendez, R. (2019). New Approach based on Autoencoders to Monitor the Tool Wear Condition in HSM. *IFAC-PapersOnLine*, 52(11):206–211.

Feldman, R. and Valdez-Flores, C. (2004). *Applied Probability and Stochastic Processes*. Thomson Brooks/Cole, 1 edition.

Fish, R., Ostendorg, M., Bernard, G., and Castanon, D. (2003). Multivel Classification of Milling Tool Wear with Confidence Estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(1):75–85.

Guo, H., Zhang, Y., and Zhu, K. (2022). Interpretable Deep Learning Approach for Tool Wear Monitoring in High-Speed Milling. *Computers in Industry*, 138(1):10638.

Haber, R. and Alique, A. (2003). Intelligent Process Supervision for Predicting Tool Wear in Machining Processes. *Int. J. Producton Research*, (13):825–849.

Hoang, T., Nguyen, Q., Nguyen, V., and Tran, N. (2020). Self-Adjusting On-line Cutting Condition for High-Speed Milling Process. *Journal of Mechanical Science and Technology*, 34(8).

Javed, K., Gouriveau, R., Zemouri, R., and Li, X. (2012). Robust, Reliable and Applicable Tool Wear Monitoring and Prognostic: Approach based on an Improved-Extreme Learning Machine. *IEEE*.

Jawahir, I. and Wang, X. (2007). Development of Hybrid Predictive Models and Optimization Techniques for Machining Operations. *J of Materials Processing Technology*, (185):46–59.

Kant, G. and Sangwan, K. (2014). Prediction and Optimization of Machining Parameters for Minimizing Power Consumption and Surface Roughness in Machining. *J of Cleaner Production*, 83:151–164.

Karabulut, S. (2015). Optimization of Surface Roughness and Cutting Force during AA7039/Al2O3 Metal Matrix Composites Milling using Neural Networks and Taguchi Method. *Measurement*, 66:139–149.

Karkalos, N., Galanis, N., and Markopoulos, A. (2016). Surface Roughness Prediction for the Milling of Ti-6Al-4V ELI Alloy with the Use of Statistical and Soft Computing Techniques. *Measurement*, 90:25–35.

Khorasani, A. and Soleymani-Yazdi, M. (2017). Development of a Dynamic Surface Roughness Monitoring System based on Artificial Neural Networks (ANN) in Milling Operation. *Int J. Adv Manuf Technol*, 93:141–151.

Lee, K., Kang, M., Jeong, Y., Lee, D., and Kim, J. (2001). Simulation of Surface Roughness and Profile in High-Speed End Milling. *Materials Processing Technology*, (113):410–415.

Monostori, L. (2000). Intelligent Machines. In *Proc of the 2nd Conf on Mechanical Eng*, pages 24–36, Hungary.

Montgomery, D. (2001). *Design and Analysis of Experiments*. John Wiley and sons, Inc., Clearence center, 222 Rosewood Drive, Danvers, MA 01923, 5 edition.

Morales-Menendez, R., Aguilar, S., Rodríguez, C., Guedea-Elizalde, F., and Garza, L. (2005). Sensor-Fusion System for Monitoring a CNC-Milling Center. *Lectures Notes in Computer Science,*, 3789:1164–1174.

Mukherjee, I. and Ray, P. (2006). A Review of Optimization Techniques in Metal Cutting Processes. *Computers and Industrial Eng*, (50):15–34.

Mursec, B. and Cus, F. (2003). Integral Model of Selection of Optimal Cutting Conditions from Different Databases of Tool Makers. *J of Material Processing Technology*, (133):158–165.

Ozcelik, B. and Bayramoglu, M. (2006). The Statistical Modeling of Surface Roughness in High-Speed Flat End Milling. *Int J of Machine Tools and Manufacture*, (46):1395–1402.

Palanisamy, P., Rajendran, I., and Shanmugasundaram, S. (2007). Optimization of Machining Parameters Using Genetic Algorithm and Experimental Validation for End-Milling Operations. *J of Material Processing Technology*, Int J. Advanced Manufacturing Technology(32):644–655.

Saglam, H. and Unuvar, A. (2003). Tool Condition Monitoring in Milling based on Cutting Forces by a Neural Network. *Int J of Production Research*, 41(7):1519–1532.

Sai, K. and Bouzid., W. (2005). Roughness Modelling in Up-Face Milling. *Int J Adv Manuf Technol*, (26):324–329.

Subramanian, M., Sakthivel, M., and Sudhakaran, R. (2014). Modeling and Analysis of surface roughness of AL7075-T6 in end milling process using Response Surface Methodology. *Arab J Sci Eng*, (39):7299–7313.

Suresh, P., Venkateswara, P., and Deshmukh, S. (2002). A Genetic Algorithm Approach for Optimization of Surface Roughness Prediction Model. *Int J. of Machine Tools and Manufacture*, (42):675–680.

Tansel, I., Ozcelik, B., Bao, W., Chen, P., Rincon, D., Yang, S., and Yenilmez, A. (2006). Selection of Optimal Cutting Conditions by Using GONNS. *Int J. of Machine Tools and Manufacture*, (46):26–35.

Tsai, Y., Chen, J., and Lou, S. (1999). An In-Process Surface Recognition System based on Neural Networks in End Milling Cutting Operations. *Machine Tools and Manufacture*, (39):583–605.

Tzeng, Y. and Chen, F. (2005). Optimization of High Speed CNC Milling Process Using Two-Phase Parameter Design Strategy by the Taguchi Methods. *JSME Int J. Series C*, 48(4):775–783.

Vallejo, A. and Morales-Menendez, R. (2010). Cost-Effective Supervisory Control System in Peripheral Milling using HSM. *Annual Reviews in Control*, 34(1):155–162.

Vallejo, A., Morales-Menendez, R., Rodriguez, C., and Sucar, L. (2006). Diagnosis of a Cutting Tool in a Machining Center. In *IEEE Int J Conf on Neural Networks*, pages 7097–7101.

Vallejo-Guevara, A., Morales-Menendez, R., and Alique, J. (2007). Designing a Cost-Effective Supervisory Control System for Machining Process. 40(19):147–152.

Vallejo-Guevara, A., Morales-Menendez, R., and Alique, J. (2008). Intelligent Monitoring and Decision Control System for Peripheral Milling Process. In *IEEE Int. Conf. on Systems, Man, and Cybernetics*, pages 1620–1625, Singapore.

Vallejo-Guevara, A., Morales-Menendez, R., and ELizalde-Siller, H. (2009a). Intelligent Control System for HSM. *J of Automation, Mobile Robotics & Intelligent Systems*, 3(3):54–63.

Vallejo-Guevara, A., Morales-Menendez, R., and Elizalde-Siller, H. (2009b). Surface Roughness Modeling in Peripheral Milling Proceses. In *NAMRC 37*.

Vallejo-Guevara, A., Morales-Menendez, R., Ramírez-Mendoza, R., and Garza-Castañon, L. (2009c). Online Prediction of Surface Roughness in Peripheral Milling Processes. In *European Control Conf*, pages 3695–3700.

Vallejo-Guevara, A., Nolazco-Flores, J., Morales-Menendez, R., Sucar, L., and Rodríguez, C. (2005). Tool-Wear Monitoring Based on Continuous Hidden Markov Models. In *Progress in Pattern Recognition, Image Analysis and Applications. Lecture Notes in Computer Science*, number 3773, pages 880–890, Heidelberg. Springer.

Wong, E. and Sridharan, S. (2001). Comparison of Linear Prediction Cepstrum Coefficients and Mel-frequency Cepstrum Coefficients for Language Identification. *Proc of 2001 Int Symp on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No.01EX489)*, pages 95–98.

Wu, T. and Lei., K. (2019). Prediction of Surface Roughness in Milling Process using Vibration Signal Analysis and Artificial Neural Network. *The Int J of Advanced Manufacturing Technology*, 102:305–314.

Yang, Y., Guo, Y., Huang, Z., Chen, N., Li, L., Jiang, Y., and He, N. (2019). Research on the Milling Tool Wear and Life Prediction by Establishing an Integrated Predictive Model . *Measurement*, 145:178–179.

Yih-Fong, T. (2005). A Hybrid Approach to Optimize Multiple Performance Characteristics of High-Speed Computerized Numerical Control Milling Tool Steels . *Materials and Design*, page 110.

Özel, T. and Karpat, Y. (2005). Predictive Modeling of Surface Roughness and Tool Wear in Hard Turning using Regression and Neural Networks . *Machine Tools and Manufacture*, (45):467–479.

Zuperl, U., Cus, F., and Kiker, E. (2006). Intelligent Adaptive Cutting Force Control in End-Milling . *Technical Gazette*, 13(1-2):15–22.

Zuperl, U., Cus, F., Mursec, B., and Ploj, T. (2004). A Hybrid Analytical- Neural Network Approach to the Determination of Optimal Conditions . *J of Materials Processing Technology*, pages 82–90.

Chapter 3
Experimental results of fault isolation approach for a 2-DOF helicopter

Manuel Alejandro Zuñiga[1], Luis Alejandro Ramírez[1], Gerardo Romero Galvan[1], Efraín Alcorta García[2], and Aldo J. Muñoz Vázquez[3]

Abstract Fault isolation increasingly becomes an important aspect for the feasibility and robustness of systems. The knowledge of a fault's presence offers important decision-making information related to the control of a system (i.e. information to keep a system in a healthy and functional state) and/or also for organizing maintenance. The case of a two degree of freedom (2-DOF) helicopter is studied. Practical aspects related to the fault modeling and implementation, in order to have the possibility of a detailed study of the used algorithm, are discussed. A previously developed model-based fault isolation algorithm is implemented by using specific hardware. A comparison between simulations and experimental results is also provided, and some directions for facing remaining challenges are also discussed.

3.1 Introduction

In recent years, fault diagnosis has been an important research topic, mainly because of its relationship with the safety and reliability of industrial processes. Fault diagnosis is useful for establishing maintenance programs for devices that are part of control systems and for developing fault-tolerant control schemes. In the last 50 years, different approaches for isolating faults in control systems have been proposed, as seen in Blanke et al. (2016), Shen et al. (2017), Varga (2017), Zhang et al. (2018), and Ding (2020). A failure in a control system is considered when a change in some parameters of the system occurs in such a way that it operates outside the tolerance margin for which it was designed. For this reason, it is important to establish mechanisms that allow determining when a fault occurs, which is known as the fault diagnosis and isolation problem.

One of the first analytical redundancy algorithms for fault diagnosis is the so-called model-based approach, i.e. they are based on mathematical models, which can be consulted in Blanke et al. (2016) and Ding (2020); here, the mathematical model plays a central role in the design of fault detection and isolation algorithms. Other approaches that have been recently proposed appear in Patan (2019), where neural networks are used primarily to determine fault-tolerant control algorithms; Concepts based on

[1]Universidad Autónoma de Tamaulipas, Unidad Académica Multidisciplinaria Reynosa Rodhe, Mexico,
e-mail: manuel.zuniga@docentes.uat.edu.mx, laramirez@docentes.uat.edu.mx and gromero@uat.edu.mx
[2]Universidad Autónoma de Nuevo León, Facultad de Ingeniería Mecánica y Eléctrica, Mexico, e-mail: efrain.alcortagr@uanl.edu.mx
[3]Texas A&M University, Department of Multidisciplinay Engineering, McAllen, TX 78504, USA, e-mail: aldo.munoz.vazquez@gmail.com

artificial intelligence have also been used to solve this problem, see Korbicz et al. (2004). Furthermore, artificial intelligence and data-driven approaches have been employed to solve this problem. See Korbicz et al. (2004) and Mansouri et al. (2020), as well as in Ding (2014) who includes some model-inspired approaches. Fault diagnosis of hybrid dynamic and complex systems has been considered in Sayed-Mouchaweh (2018). Furtermore an adaptive techniques for fault diagnosis are discussed in Shen et al. (2017), and Wang et al. (2020) consider iterative learning approaches for fault diagnosis. The diagnosis of fractional-order systems is studied in Martinez-Guerra et al. (2021), and fault diagnosis in switched systems is revised in Du et al. (2021). Among the above mentioned books considering principally theoretical results, there are also references that consider specific applications, such as Zolghadri et al. (2014) for fault diagnosis, fault-tolerant control, and the guidance of aerospace vehicles. Other aspects such as the fault detection of wind turbine systems have been considered in Reza Karimi (2018). Induction motors fault diagnosis is discussed in Karmakar et al. (2016). Model-based condition monitoring, actuator, sensors, machinery, plants, and drives are considered in Isermann (2011).

This work presents an analysis and design of a fault isolation scheme for a Quanser® 2-DOF Helicopter. Mainly, a Hamiltonian model of the system previously mentioned is derived. Faults in the system are also modeled as well as implementation methods (with the results of this study). A fault isolation scheme is derived using nonlinear decoupling and observer-based residual generation, as in Ramírez et al. (2020). The implementation of the fault isolation algorithms, as well as the experimental results, are also discussed. Some practical experiences related to the implementation are presented.

The rest of this work is organized as follows: the next section shows some preliminary results that will be used later in the paper. The system under consideration is presented in Section 3, and details about the design of the fault isolation scheme are considered in Section 4. The experimental results are presented in Section 5, and, finally, some conclusions are summarized in Section 6.

3.2 Preliminaries

3.2.1 Generalized Hamiltonian systems

A class of nonlinear systems admits a representation in a generalized Hamiltonian form, as described by Rodriguez-Alfaro et al. (2015), Sira Ramírez and Cruz Hernández (2001) and Van der Schaft (2017). It is given by

$$\begin{aligned} \dot{x} &= [J(x)+S(x)]\frac{\partial H(x)}{\partial x}+F(x)+Gu \\ y &= C\frac{\partial H(x)}{\partial x} \end{aligned} \tag{3.1}$$

with the state vector $x \in \Re^n$, the input vector $u \in \Re^m$, $y \in \Re^p$ is the output vector, $G \in \Re^{n\times m}$ being a constant matrix, $F \in \Re^n$ a vector of nonlinear functions and $J \in \Re^{n\times n}$ being a skew-symmetric interconnection matrix ($J(x) = -J^T(x)$) associated with a simple bilinear form, which represents the conservative part of the system. $S(x) \in \Re^{n\times n}$ is a symmetric matrix ($S(x) = S^T(x)$), $C \in \Re^{p\times n}$ is an output constant matrix . $H(x)$ is a smooth vector-valued function of and real values, and it can be understood as the

generalized energy defined as

$$H(x) = \frac{1}{2} x^T M x \tag{3.2}$$

where M is a constant matrix, symmetric and positive definite, and $\partial H(x)/\partial x$ represents the gradient of $H(x)$. Finally, $F(x)$ is a vector field that contains the non-linear parts of the system.

3.2.2 Fault isolation schema

The fault isolation strategy utilized in this paper was presented in Ramírez et al. (2020); it is basically a bank of nonlinear observer-based residual generators. Each residual generator is designed for a decoupled subsystem so that each subsystem is associated with a set of faults. Moreover, each subsystem is obtained by taking advantage of the specific model structure of the 2-DOF helicopter.

In addition, the decoupling of the effect of some specific faults could be achieved by using the output measurements, which are linear with respect to the state. This will be made clear later. Roughly speaking, the residuals can be represented as shown in Fig. 3.1.

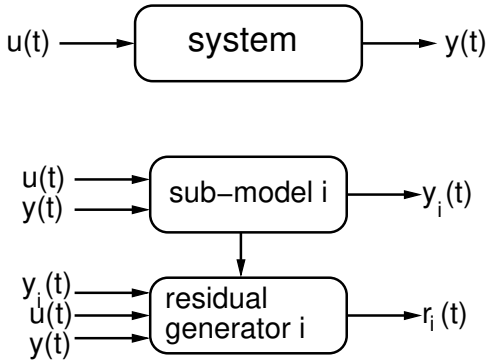


Fig. 3.1: Schematic of residuals

3.3 Helicopter with two degrees of freedom

The system used in this work is the Quanser® 2-DOF helicopter, which is a lab test prototype standing on a fixed pedestal (that pivots) and consists of two motors: one on the front for the lift movement and one rear or tail motor for the yaw movement. See Fig. 3.2.

The system model is represented by means of the following nonlinear equation, in vector form:

$$D(q)\ddot{q} + N(q,\dot{q}) + g_v(q) = \tau, \tag{3.3}$$

where $\tau \in \Re^2$ represents the input vector, $g_v(q) \in \Re^2$ is the vector of gravitational pairs, $D(q) \in \Re^{2\times 2}$ corresponds to the inertial matrix, which is symmetric and positive definite, $N(q,\dot{q}) \in \Re^{2\times 2}$ models the

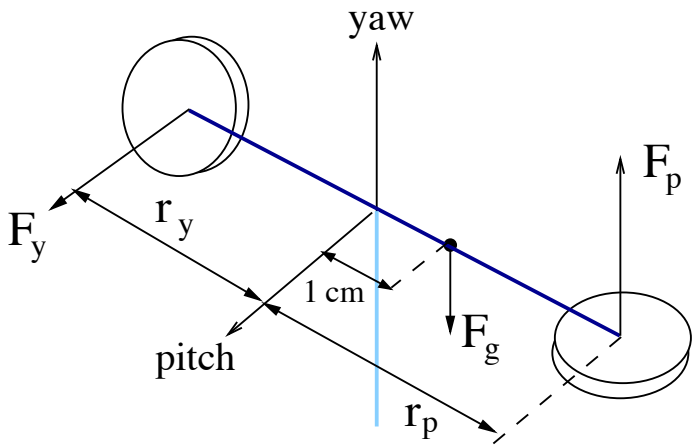


Fig. 3.2: Schematic of the Quanser® 2-DOF helicopter

velocity quadratic effects generated by centrifugal and Coriolis forces. For the Quanser® system, define the generalized coordinate vector $q = [\phi, \psi]^T$, where ϕ and ψ are the pitch and yaw angles. Consider $\tau = [U_p, U_y]^T$, and

$$D(q) = \begin{bmatrix} m_1 {l_1}^2 + m_2 {l_2}^2 & 0 \\ 0 & m_1 {l_1}^2 \cos^2(\phi) + m_2 {l_2}^2 \cos^2(\phi) \end{bmatrix},$$
$$g_v(q) = \begin{bmatrix} m_1 g l_1 \cos(\phi) - m_2 g l_2 \cos(\phi) \\ 0 \end{bmatrix}.$$

3.3.1 Hamiltonian representation

Hamiltonian representation brings a structure to a model of a dynamic system and this will be used later to design a fault detection and isolation scheme.

A way to obtain the Hamiltonian representation from the Euler-Lagrange representation (3.3) (as in Van der Schaft (2017), for example) is by computing the generalized moment $p(t)$, defined by

$$p = D(q)\dot{q} \tag{3.4}$$

and the Hamiltonian function

$$\mathscr{H}(q,p) = \frac{1}{2} p^T D^{-1}(q) p + U(q), \tag{3.5}$$

where $U(q)$ represents the potential energy of the system. Thus, the system´s model (3.3) can be alternatively written as

$$\dot{q} = \frac{\partial \mathscr{H}(q,p)}{\partial p} = D^{-1}(q)p, \tag{3.6}$$

$$\dot{p} = -\frac{\partial \mathscr{H}(q,p)}{\partial q} + \tau = -\frac{\partial}{\partial q}\left(\frac{1}{2}p^T D^{-1}(q)p\right) - \frac{\partial}{\partial q}U(q) + \tau, \tag{3.7}$$

where $\frac{\partial}{\partial q}U(q) = g_v(q)$.

The system equations that represent (3.3) are the following:

$$\dot{x}_1 = \frac{x_3}{l^2(m_1+m_2)}, \tag{3.8}$$

$$\dot{x}_2 = \frac{x_4}{l^2\cos^2(x_1)(m_1+m_2)}, \tag{3.9}$$

$$\dot{x}_3 = \frac{-x_4^2\sin(x_1)}{l^2\cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2) + U_p, \tag{3.10}$$

$$\dot{x}_4 = U_y \tag{3.11}$$

where the system state vector is given by $x^T = [x_1 \quad x_2 \quad x_3 \quad x_4]$ with $x_1 = \phi$, $x_2 = \psi$, $x_3 = \dot{\phi}$ and $x_4 = \dot{\psi}$. The control law $\tau = [U_p, U_y]^T$ is a proportional derivative (PD) type controller, and it was tuned using a procedure presented in Romero et al. (2012). The variables and parameters used in the previous equations are as follows: $x_1(\phi)$ is the pitch angle; $x_2(\psi)$ is the yaw angle; $x_3(\dot{\phi})$ corresponds to the pitch angular velocity; $x_3(\dot{\phi})$ represents the pitch angular velocity; $x_4(\dot{\psi})$ gives the yaw angular velocity; U_p is the pitch control input; U_y corresponds to the yaw control input; m_1 represents the mass of lift motor; m_2 is the mass of the tail motor; l_1 corresponds to the length m_1 to center of mass; l_2 is the length m_2 to center of mass and g represents the gravity.

The following generalized Hamiltonian function is defined: $H = \frac{1}{2}x^T x$, with $M = I_{4\times 4}$. H is the basis of the generalized Hamiltonian representation

$$\dot{x} = J(x)\frac{\partial H(x)}{\partial x} + S(x)\frac{\partial H(x)}{\partial x} + \begin{bmatrix} \frac{x_3}{l^2(m_1+m_2)} \\ \frac{x_4}{l^2\cos^2(x_1)(m_1+m_2)} \\ \frac{-x_4^2\sin(x_1)}{l^2\cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} U_p \\ U_y \end{bmatrix}}_{u}, \tag{3.12}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \frac{\partial H(x)}{\partial x}, \tag{3.13}$$

where, $\frac{\partial H}{\partial x} = x$, $J(x) = [0] \in \mathbb{R}^{4\times 4}$, $S(x) = [0] \in \mathbb{R}^{4\times 4}$.

3.3.2 Fault Modeling

Four possible faults are studied in this work: one for each sensor and one for each actuator. In general, faults could be additive or multiplicative; however, in this paper the focus is on the case of additive faults, which are modeled as exogenous inputs to the system. Note that faults can be also incipient, abrupt or

intermittent, depending on the form in which they are manifested. See for example Chen and Patton (1999), Ding (2013) and Isermann (2011).

3.3.2.1 Additive faults' representation

As mentioned above, additive faults are modeled with additive terms in the actuators channels f_{ai}, as well as in the sensors channels f_{si}, with $i = 1,2,3$. The representation of the system´s model with faults (sensor and actuator) is given as follows:

$$\dot{x} = J(x)\frac{\partial H(x)}{\partial x} + S(x)\frac{\partial H(x)}{\partial x} + F(x) + Gu + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{E_f} \begin{bmatrix} f_{a1} \\ f_{a2} \end{bmatrix}, \tag{3.14}$$

$$y = C\frac{\partial H(x)}{\partial x} + \begin{bmatrix} f_{s1} \\ f_{s2} \end{bmatrix}. \tag{3.15}$$

3.3.2.2 Fault decoupling

Sensor faults

In order to obtain two sub-models where each of them is associated with only one sensor fault, the following strategy is used. Firstly, the output related to sensor one (output one) is considered; here, the corresponding differential equation (3.8) of the states required to integrate output one is considered as part of the sub-model one. Additionally, the differential equations of the states appearing in the previous differential equations, i.e. equations (3.9), (3.10) and (3.11), should be included in the sub-model as well.

Secondly, referring to sub-model two, output two (related to sensor two) is considered. The corresponding differential equations of the states required to integrate output two are considered as part of the sub-model two. As in the first case, the differential equations of the states appearing in the previous differential equations should also be included in the sub-model.

Sub-model one. For the first sensor, only the state x_1 is involved, so the differential equation for $\dot{x}_1$ is required first. Because, x_3 appears in the equation of $\dot{x}_1$, however, the differential equation for $\dot{x}_3$ should also be included in the sub-model. Note that x_4 is present in the differential equation of $\dot{x}_3$, so the differential equation for $\dot{x}_4$ is also required in sub-model one:

$$\dot{x}_1 = \frac{x_3}{l^2(m_1+m_2)}, \tag{3.16}$$

$$\dot{x}_3 = \frac{-x_4^2\sin(x_1)}{l^2\cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2) + U_p + f_{a1}, \tag{3.17}$$

$$\dot{x}_4 = U_y + f_{a2}, \tag{3.18}$$

$$y_{s1} = x_1 + f_{s1} \tag{3.19}$$

Note that in sub-model one, the fault of sensor two (f_{s2}) is not affecting, i.e. it is not present in the submodel one.

Sub-model two. For the second subsystem, only the second output is considered. The state variable involved is x_2, so $\dot{x}_2$ should be included. In the differential equation of $\dot{x}_2$, x_1 and x_4 are also present; therefore, $\dot{x}_1$ and $\dot{x}_4$ are included. Finally, x_3 is also present, so $\dot{x}_3$ is also included:

$$\dot{x}_1 = \frac{x_3}{l^2(m_1+m_2)}, \tag{3.20}$$

$$\dot{x}_2 = \frac{x_4}{l^2\cos^2(x_1)(m_1+m_2)}, \tag{3.21}$$

$$\dot{x}_3 = \frac{-x_4^2\sin(x_1)}{l^2\cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2)+U_p+f_{a1}, \tag{3.22}$$

$$\dot{x}_4 = U_y + f_{a2}, \tag{3.23}$$

$$y_{s2} = x_2 + f_{s2} \tag{3.24}$$

Note that in the sub-model two, the fault of sensor one (f_{s1}) is not affecting, i.e. it is not present in the submodel two.

Actuator faults

Similar to the case of sensor faults, we obtained the decoupled subsystems that are sensitive only to one particular fault. Thus, the corresponding residual is generated from an observer based on output feedback.

Sub-model three. The first actuator fault (f_{a1}) is connected to the differential equation of the state variable x_3 of U_p since x_1 is known and x_2 is assumed to be free of fault.

$$\dot{x}_1 = \frac{x_3}{l^2(m_1+m_2)}, \tag{3.25}$$

$$\dot{x}_3 = \frac{-\left(\dot{x}_2 l^2\cos^2(x_1)(m_1+m_2)\right)^2\sin(x_1)}{l^2\cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2)+U_p+f_{a1}, \tag{3.26}$$

$$y_1 = x_1 + f_{s1}. \tag{3.27}$$

Note that x_4 is not available, but it can be obtained from the dynamics of the system as

$$x_4 = \dot{x}_2 l^2\cos^2(x_1)(m_1+m_2), \tag{3.28}$$

and $\dot{x}_2$ is obtained through numerical differentiation from the measurement: $y_2 = x_2 + f_{s2}$.

Sub-model four. The differential equation for $\dot{x}_4$ is now considered. Note that no additional state variable is strictly required, however, in order to connect with the output the differential equation because $\dot{x}_2$ is also included in the model together with the second output:

$$\dot{x}_2 = \frac{x_4}{l^2\cos^2(x_1)(m_1+m_2)} = \frac{x_4}{l^2\cos^2(y_1-f_{s1})(m_1+m_2)}, \tag{3.29}$$

$$\dot{x}_4 = U_y + f_{a2}, \tag{3.30}$$

$$y_2 = x_2 + f_{s2}. \tag{3.31}$$

in addition x_1 is obtained from the first output equation as: $x_1 = y_1 - f_{s1}$

Remark 1. Note that for fault decoupling an inspection procedure is used instead of a nonlinear transformation, in other words, for sensor *i*, the output is selected (omitting all other outputs) as well as all the differential equations associated with the states that explicitly appear in the selected output and the ones required to complete the states present in the differential equations selected. In Table 3.1 a summary of the sensitivity of each of the different faults is presented, where $\surd$ means that the sub-model is sensitive to the corresponding fault.

The fault incidence matrix is as follows:

Table 3.1: Sensitivity of sub-models with respect to faults

sub-models	f_{s1}	f_{s2}	f_{a1}	f_{a2}
1	$\surd$	0	$\surd$	$\surd$
2	0	$\surd$	$\surd$	$\surd$
3	$\surd$	$\surd$	$\surd$	0
4	$\surd$	$\surd$	0	$\surd$

where $\surd$ means that the corresponding sub-model is affected (at least theoretically) by the respective fault. In contrast, 0 means that the sub-model is decoupled from the respective fault.

3.4 Fault isolation for a 2-DOF Helicopter

3.4.1 Residual Generator Design

The observer-based residuals are built following the method presented in Ramírez et al. (2020). It takes advantage of the Hamiltonian structure of the model, and it is rewritten here for completeness:

$$\dot{x} = [J(y) + S(y)]\frac{\partial H(x)}{\partial x} + F(x) + Gu, \tag{3.32}$$

$$y = C\frac{\partial H(x)}{\partial x}. \tag{3.33}$$

The state x of the nonlinear system (3.1) is estimated by the system

$$\dot{\hat{x}} = [J(y) + S(y)]\frac{\partial H(\hat{x})}{\partial \hat{x}} + F(\hat{x}) + Gu + K(y-\eta), \tag{3.34}$$

$$\eta = C\frac{\partial H(\hat{x})}{\partial \hat{x}}, \tag{3.35}$$

if the pair (C,S) is observable or at least detectable and the matrix

$$\Xi = M^T \left[S - \frac{1}{2}(KC + C^T K^T) \right] M + \Pi \tag{3.36}$$

with $\Pi = \frac{1}{2}\left[M \frac{\partial F(x)}{\partial x} + \frac{\partial F(x)}{\partial x}^T M^T \right]$ is negative definite.

For the construction of the bank of residual generators, an observer for each sub-model should be designed. Considering the sub-model one, which has been obtained to have the effect of the sensor of fault one (f_{s1}), and the analysis made, the actuator faults f_{a1} and f_{a2} also affect sub-model one:

$$\dot{x}_1 = \frac{x_3}{l^2}(m_1 + m_2), \tag{3.37}$$

$$\dot{x}_3 = \frac{-x_4^2 \sin(x_1)}{l^2 \cos^3(x_1)(m_1 + m_2)} - gl\cos(x_1)(m_1 - m_2) + U_p, \tag{3.38}$$

$$\dot{x}_4 = U_y + f_{a2}, \tag{3.39}$$

$$y_{s1} = x_1 + f_{s1}. \tag{3.40}$$

leads to its generalized Hamiltonian representation, with

$$\frac{\partial H}{\partial x} = [x_1 \quad x_3 \quad x_4]^T. \tag{3.41}$$

The measurement of the output variables was performed by two rotary optical encoders: one placed at the base of the platform, which allows the yaw angle to be measured ψ, and the other placed in the center of the platform, which measures the pitch angle ϕ. See Fig 3.3.

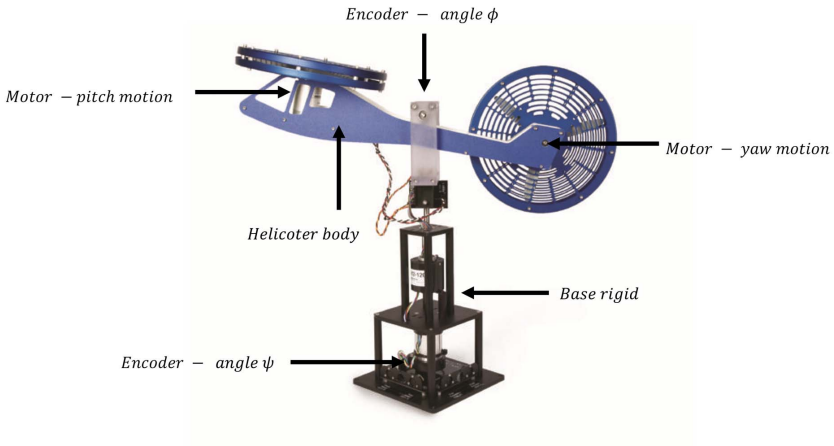


Fig. 3.3: Helicopter structure taken from Quanser Inc. (2012)

The parameters of the previous model are the following:

$$J(x) = [0] \in \mathbb{R}^{3x3}, \quad S(x) = [0] \in \mathbb{R}^{3x3}, \quad F(x) = \begin{bmatrix} \frac{x_3}{l^2(m_1+m_2)} \\ \frac{-x_4^2 \sin(x_1)}{l^2 \cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2) \\ 0 \end{bmatrix},$$

$$G = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1\,0\,0 \end{bmatrix}.$$

The pair (C,S) is detectable according to Definition 2.1 in Sira Ramírez and Cruz Hernández (2001) since the matrix $\begin{bmatrix} C \\ \lambda I_{3\times 3} - S \end{bmatrix}$ is of full rank for any value of λ in $\mathbb{Z}_{<0} = \{\bar{\lambda} \in \mathbb{Z} : \Re\{\bar{\lambda}\} < 0\}$. Thus, the residual is defined from the observer:

$$\begin{aligned} \dot{\hat{x}}_1 &= \frac{\hat{x}_3}{l^2(m_1+m_2)} + L_1(y_1 - \hat{x}_1), \\ \dot{\hat{x}}_3 &= \frac{-\hat{x}_4^2 \sin(x_1)}{l^2 \cos^3(x_1)(m_1+m_2)} - gl\cos(x_1)(m_1-m_2) + U_p, \\ \dot{\hat{x}}_4 &= U_y, \\ \hat{y}_{s1} &= \hat{x}_1. \end{aligned}$$

Note that the residual is designed from a copy of the original subsystem and the correction factor $L_1(y_1 - \hat{y}_1)$.

For the fault in the second sensor (f_{s2}) and those in the two actuators (f_{a1}) and (f_{a2}), the decoupling is similar to the procedure utilized in the fault (f_{s1}).

In Table 1, we can see that if single faults are occurring, they can be isolated. The four residuals are obtained assuming that only one fault at a time is present and not all simultaneously. In addition, a residual is obtained considering a derivative in time, so obtaining this derivative could be a difficult task in the presence of noise. It is important to notice that the generation of residuals based on observers constitutes a reliable option for reducing noise effects because of the inherent filtering characteristic of the observer; see Rodriguez-Alfaro et al. (2015).

3.5 Experimental results

The 2-DOF experimental platform consists of a helicopter´s body mounted on a rigid metal base. It has two motors mounted perpendicularly to each other. This emulates the typical helicopter configuration with a main rotor, which generates the pitch motion, and a tail rotor that produces the yaw motion. DC motors work with a nominal voltage of 12 V. The platform parameters are shown in Table 3.2.

The measurement of the output variables was carried out by two rotary optical encoders: one placed at the base of the platform, which allows the yaw angle to be measured ψ, and the other placed in the center of the platform, which measures the pitch angle ϕ (3.3).

Table 3.2: The platform parameters

Parameter	Description	Value
m_1	Mass of lift motor	$0.014kg$
m_2	Mass of tail motor	$0.232kg$
1_1	Length m_1 to center of mass	$0.203m$
1_2	Length m_2 to center of mass	$0.203m$
g	Gravity	$9.8m/s^2$

To implement the algorithm, the Simulink© tool from MATLAB© was used. Data acquisition was performed using an Arduino Mega 2560, which consists of a 54 input/output pins, including 14 pins with the ability to work as pulse width modulation and 16 pins as analog inputs, a microprocessor, a 16 MHz crystal, and a 256K flash memory, with a working range between 7 V and 12 V. In the power part, a SparkFun Monster Moto Shield was used, with a capacity of 16 V, and a current of 30 A, 20 kHz of PWM frequency.

It is important to mention that the faults produced in each of the physical tests were in a proportion of 50% with respect to the output, and they were presented as abrupt faults.

Introduced sensor faults

They are considered abrupt failures in the sensors; they appear suddenly and affect the measured signal of the encoders. As a result, the signal used to read the pitch and yaw angles, which is later fed back to the control algorithm, presents deviations between estimated and the current (real) angle of the helicopter position.

To produce a failure in the sensors, we considered adding an external signal to the sensor´s signal that feeds the algorithm. The signal is formed virtually from a block of noise-limited according to a percentage of the failure with respect to the output.

Introduced actuator faults

Helicopter motors are considered actuators that convert electrical energy into rotary motion. Propellers generate thrust force that produces displacement in pitch and yaw, respectively. Because of the aerodynamic profile of the propellers, when rotating, a difference in speed is obtained on each side of the blade between the fluid on one side and the other. Therefore, the difference in speeds leads to a difference in pressure, and, thus, a force that is perpendicular to the plane of rotation is generated, which is known as the propulsion force. Once the helicopter propeller begins to rotate, it picks up the air and is an accelerator. The speed of the air behind the disk, multiplied by the mass of displaced air, gives us the thrust force. Based on this idea and in order to cause a decrease in the thrust force of the helicopter, different ways of blocking the air (like covers) surrounding the helicopter were designed that correspond to 50% of the area of the rotating disk, which, when placed on the propellers, reduced the airflow, causing the

helicopter to lose thrust and lift. These covers were placed abruptly in order to have the same effect as a partial actuator failure.

3.5.1 Fault scenarios and experimental results

In order to show the effectiveness of the designed residuals, four cases are presented, one for each considered fault. For each case, the four residuals obtained from the experimental platform are shown.

Fault in sensor one

A 50% change in the sensor signal of the output one is considered, i.e., of the measured angle ϕ. The change occurs 15 seconds after the initial time.

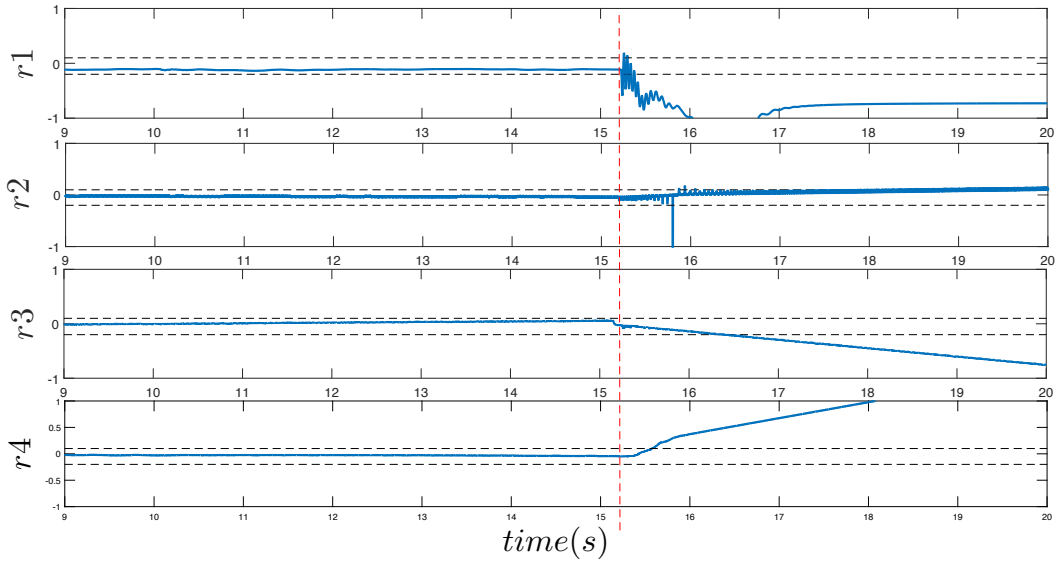


Fig. 3.4: Residuals' response to sensor fault one (f_{s1})

Fault in sensor two

A 50% change in the sensor signal of output two is considered, i.e., of the measured angle ψ. The change occurs 15 seconds after the initial time.

Fault in actuator one

An approximated 50% change in the first actuator signal is considered. The change occurs 15 seconds after the initial time approximately.

Fault in actuator two

An approximated 50% change in the second actuator signal is considered. The change occurs approximately 15 seconds after the initial time.

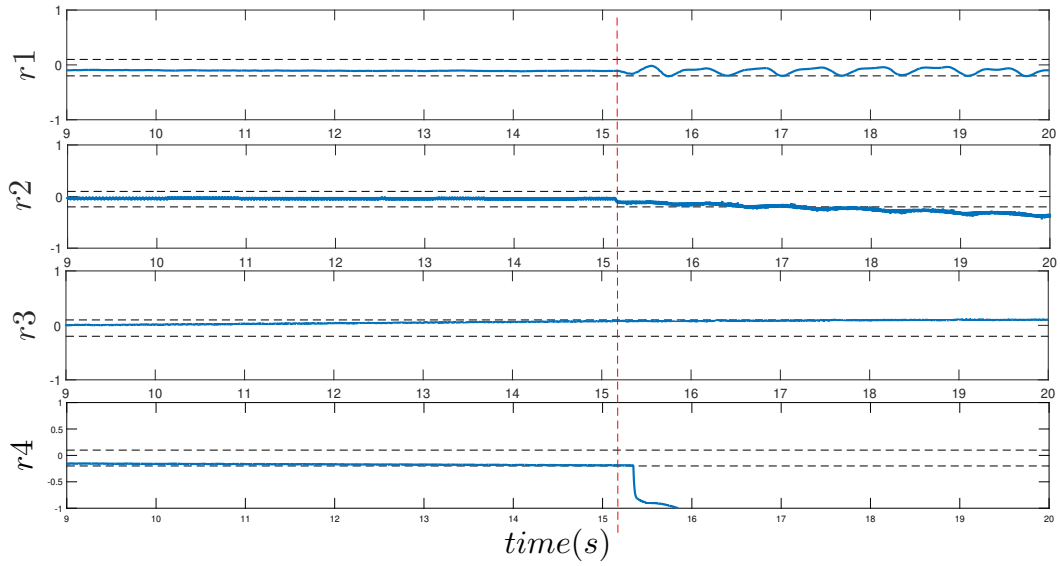


Fig. 3.5: Residuals' response to sensor fault two (f_{s2})

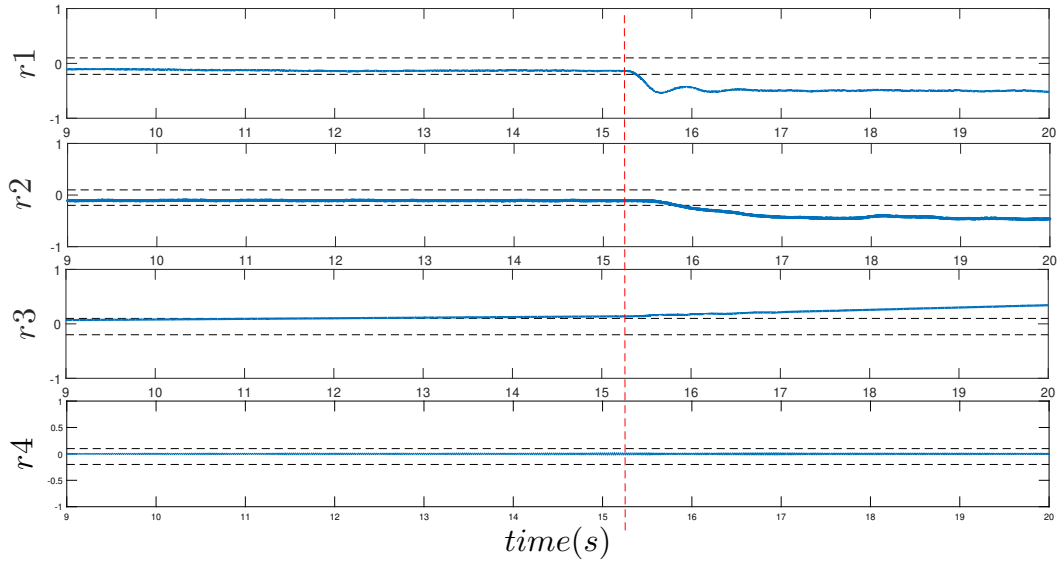


Fig. 3.6: Residuals' responses to an actuator one fault (f_{a1})

3.6 Conclusion

Model-based fault isolation algorithms represent an interesting alternative for diagnosis in many processes. Fault isolation should be considered from the beginning of the design processes in order to obtain a better result. Because the considered faults correspond to the sensor and actuator, a simple procedure was used in order to obtain dynamic redundant relations that can be used for the residual design. Considering a class of port-Hamiltonian nonlinear systems, a systematic observer-based procedure for residual design was considered. The designed residuals were tested in a simulation and afterwards were imple-

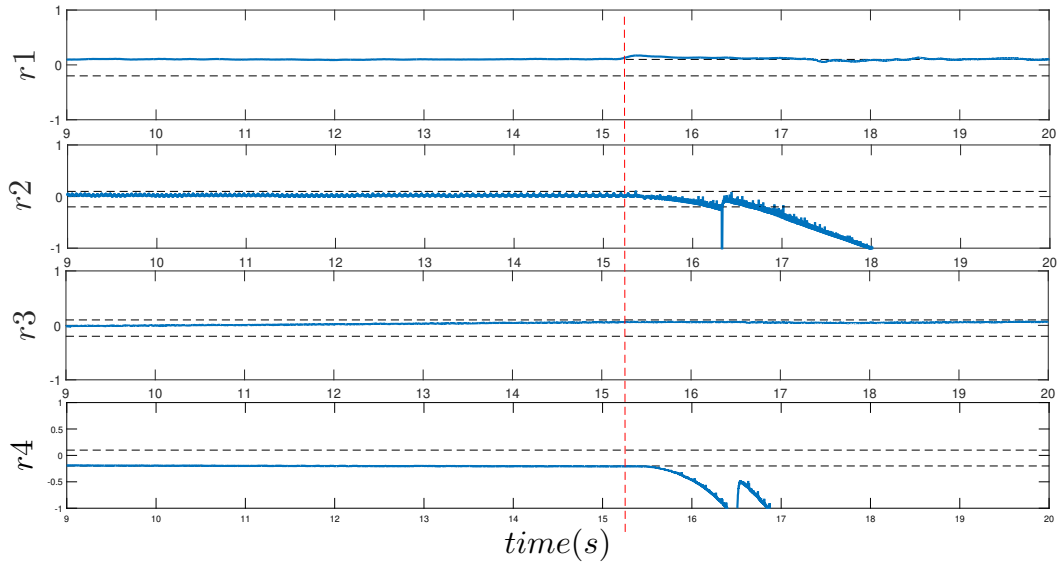


Fig. 3.7: Residuals' responses to an actuator two fault (f_{a2})

mented to be tested in a real-time embedded system. Fault isolation was obtained for the four single faults considered, in which one case also required a specific residual evaluation algorithm (how the sensor fault two, residual one, which should have been zero, but it had some reaction). An important aspect was how the fault was implemented because it allowed us to make a repetitive experiment. Some topics that could be part of future work include seeking to reduce the thresholds for fault detection and working, in this way, with small magnitude faults. On the other hand, it is essential to reduce fault detection time and isolate coupled faults. Finally, it is also of great interest to propose control strategies that allow the system to be tolerant when a fault occurs.

References

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2016). *Diagnosis and Fault-Tolerant Control*. Springer.

Chen, J. and Patton, R. J. (1999). *Robust model based fault diagnosis for dynamic systems*. Kluwer Academic Publishers Group.

Ding, S. X. (2013). *Model-based fault diagnosis techniques 2nd edition.* Springer, 2nd edition.

Ding, S. X. (2014). *Data-driven Design of Fault Diagnosis and Fault-tolerant Control Systems*. Advances in Industrial Control. Springer, London.

Ding, S. X. (2020). *Advanced methods for fault diagnosis and fault-tolerant control*. Springer.

Du, D., Xu, S., and Cocquempot, V. (2021). *Observer-Based Fault Diagnosis and Fault-Tolerant Control for Switched Systems*. Springer.

Isermann, R. (2011). *Fault-Diagnosis Applications*. Springer.

Karmakar, S., Chattopadhyay, S., Mitra, M., and Sengupta, S. (2016). *Induction Motor Fault Diagnosis*. Springer.

Korbicz, J., Koscielny, J. M., Kowalczuk, Z., Cholewa, W., and (Eds.) (2004). *Fault Diagnosis*. Springer.

Mansouri, M., Harkat, M.-F., Nounou, H. N., and Nounou, M. N. (2020). *Data-driven and model-based methods for fault detection and diagnosis*. Elsevier.

Martinez-Guerra, R., Meléndez-Vázquez, F., and Trejo-Zúñiga, I. (2021). *Fault-tolerant control and diagnosis for integer and fractional-order systems*. Springer.

Quanser Inc. (2012). *Laboratory Guide, 2 DOF Helicopter Experiment for MATLAB®/Simulink® Users*. Quanser Inc., Markham, Ontario, Canada, 1. edition.

Patan, K. (2019). *Robust and Fault-Tolerant Control: Neural-Network-Based Solutions*. Springer.

Ramírez, L. A., Zuñiga, M. A., Romero, G., Alcorta-García, E., and Muñoz-Vázquez, A. J. (2020). Fault diagnosis for a class of robotic systems with application to 2-dof helicopter. *Applied Sciences*, 10(23).

Reza Karimi, H. (2018). *Structural control and fault detection of wind turbine systems*. The Institution of Engineering and Technology.

Rodriguez-Alfaro, L. H., Alcorta-Garcia, E., Lara, D., and Romero, G. (2015). Hamiltonian approach to fault isolation in a planar vertical take–off and landing aircraft model. *Int. J. Appl. Math. Comput. Sci.*, 25:65–76.

Romero, G., Alcorta-García, E., Lara, D., Pérez, I., Betancourt, B., and Ocampo, H. (2012). New method for tuning robust controllers applied to robot manipulators. *International Journal of Advanced Robotic Systems*.

Sayed-Mouchaweh, M., editor (2018). *Fault diagnosis of hybrid dynamic and complex systems*. Springer.

Shen, Q., Jiang, B., and Shi, P. (2017). *Fault Diagnosis and Fault-Tolerant Control Based on Adaptive Control Approach*. Springer.

Sira Ramírez, H. and Cruz Hernández, C. (2001). Synchronization of chaotic systems: a generalized hamiltonian systems approach. *International journal of bifurcation and chos*, 11:1381–1395.

Van der Schaft, A. (2017). *L2-gain and passivity techniques in nonlinear control*. Springer.

Varga, A. (2017). *Solving Fault Diagnosis Problems*. Springer.

Wang, L., Zhang, R., and Gao, F. (2020). *Iterative Learning Stabilization and Fault-Tolerant Control for Batch Processes*. Springer.

Zhang, K., Jiang, B., Shi, P., and Cocquempot, V. (2018). *Observer-Based Fault Estimation Techniques*. Springer.

Zolghadri, A., Henry, D., Cieslak, J., Efimov, D., and Goupil, P. (2014). *Fault Diagnosis and Fault-Tolerant Control and Guidance for Aerospace Vehicles, From Theory to Application*. Springer.

Chapter 4
Buck-boost converter fault diagnosis for an EDF-100 distillation pilot plant

Mario Heras Cervantes[1], Adriana del Carmen Téllez Anguiano[1], and Juan Anzurez Marin[2]

Abstract Distillation is the process of separating chemical mixtures most commonly used in industry, with applications ranging from cosmetic and pharmaceutical to petrochemical industries. The equipment used to perform the distillation process is the distillation column. Initial investment and maintenance costs for distillation columns are very high; therefore, it is necessary to have an appropriate fault detection system that allows improving the safety and security of the diverse parts of the column including the heating subsystem, which generates the caloric power necessary to evaporate the mixture. This work presents a fault detection and diagnosis algorithm for the heating subsystem that is implemented by using a buck-boost converter and by using the Takagi-Sugeno fuzzy model. Practical considerations related to the implementation, analysis and a comparison of the results of the convert obtained from simulated and experimental data in a distillation column using a binary mixture (of ethanol and water) are presented.

4.1 Introduction

The dependence of modern society on technological systems and processes has increased in recent years; therefore, the proper functioning of these systems has become a necessity. On the other hand, industrial systems and processes are increasingly sophisticated because of their components and the functions that they implement. This increases their vulnerability to faults, however (Verde et al., 2013; Kordestani et al., 2019; Demidova et al., 2021).

Generally, a fault is an undesired variation in the normal behavior of the system, causing damage to the equipment and risks for the user as well as products with undesired characteristics (Verde et al., 2013).

The vast majority of control systems do not consider factors such as the malfunction of sensors, actuators or other components that can cause inadequate behavior of the system and instability or risks for the users. Therefore, industrial processes need to implement feedback and automation devices that allow better performance and greater safety (Zhong et al., 2017; Bahreini et al., 2021; Iqbal et al., 2019).

In recent years, the design of fault detection and isolation (FDI) systems has been proposed in order to detect faults and maintain stability and desired performance. Reliable, timely and efficient fault detection

[1]Tecnológico Nacional de México/Instituto Tecnológico de Morelia, Av. Tecnológico 1500, Col. Lomas de Santiaguito, Morelia, Michoacán, Mexico, e-mail: mario.hc@morelia.tecnm.mx and adriana.ta@morelia.tecnm.mx

[2]Universidad Michoacana de San Nicolás de Hidalgo, Santiago Tapia 403, Centro, Morelia, Michoacán, Mexico e-mail: juan.anzurez@umich.mx

can prevent risks for both the process and the user, which is why fault detection techniques in industrial systems have become indispensable.

At an industrial level, various processes require an FDI system to operate properly. In the particular case of the chemical industry, distillation is a process that can benefit from these systems since it requires a large amount of energy to heat the mixture. The malfunction of a component can affect the safety of the operators or the quality of the product, hence the need to implement constant monitoring techniques to avoid faults in the process (Ming and Zhao, 2017; Khan et al., 2020; Ankur et al., 2020).

In a distillation column, the process of separating one or more components of a mixture is performed from their difference in volatility, which requires an actuator to generate the caloric power necessary to evaporate the mixture. The most common actuators for this purpose are based on power electronics converters that regulate the voltage or current in the electric heating resistance responsible for generating this caloric power.

In this chapter, an FDI system applied to a buck-boost converter was developed. This converter regulates the heating power, Q_b, through the electrical power, W, in a heating resistance through the duty cycle, d, of a Pulse Width Modulation power signal in an EDF-1000 distillation pilot plant. For the sake of contextualization, the distillation column model is described in Section 4.2. The developed FDI system is composed of two fuzzy observers with sliding modes that estimate the output voltage of the converter (v_C) and the current in the inductor (i_L). With the estimation errors of each observer, the residuals r_1 and r_2 are generated, respectively, to determine the symptoms that indicate the presence or absence of a fault. The system validation is performed in simulation and uses experimental data in real-time on an EDF-1000 distillation pilot plant.

4.2 Model of a distillation column and its heating actuator

In industry, distillation is the process most commonly used to separate chemical mixtures, with the petrochemical (production of petroleum derivatives) and food (production of alcoholic beverages) industries being the most important, because of the current lifestyle of people (Ibrahim et al., 2018).

The objective of distillation is to separate two or more elements from a mixture, where the most volatile element is obtained as a distilled product. The equipment to carry out the distillation process is the distillation column or the distillation pilot plant, as shown in Fig. 4.1, which is composed of a condenser, a boiler and the body of the column consisting of multiple perforated plates.

The boiler is the element that provides the necessary heat for evaporating the liquid mixture contained in it. The vapor flow, as it moves up the plates of the column body, is enriched by the light element, i.e., the element with the lowest boiling point in the mixture. The vapor that reaches the condenser is condensed and, according to the state of the reflux valve, is extracted as a distilled product or re-enters the column. The liquid that re-enters through the reflux action descends by gravity into the body of the column, enriched with the heavy element, i.e., the element with the highest boiling point. Each plate of the distillation column corresponds to a degree of purity of the light element known as molar fraction (Téllez, 2010). Because of the cost of the required sensors or meters, the mole fraction is measured off-line.

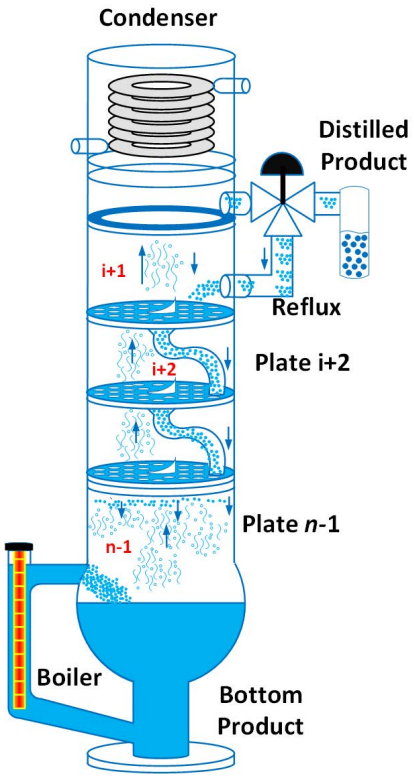


Fig. 4.1: Distillation column schematic

Fractional distillation is used to separate homogeneous liquid mixtures where the components have a difference between their boiling points of less than 25 ^oC. Generally, there are two operation modes of fractional distillation, named continuous and batch. In the continuous mode, the feeding of the liquid mixture and the extraction of the distilled product is performed continuously.

In batch distillation, the mixture is deposited in the boiler. At the end of the process, the distillate and bottom product are extracted; the batch operation mode is mainly used to separate small amounts of mixture, obtain different qualities of the distilled product for the same mixture or separate multicomponent mixtures.

A batch distillation column is not operated with constant parameters, but rather the control actions are continuously adjusted according to the state of the distillation. Therefore, the continuous and correct monitoring and control of all the variables of the process are essential for improving the quality and quantity of the distilled product, as well as the safety of the process and the users. To achieve this objective, it is necessary to have models and apply the design of observers and FDI systems.

The actuators in a distillation column have a very important role because they can modify physical variables of the process such as temperature and pressure (Paraschiv and Olteanu, 2015), by modifying the purity of the product from reflux (Alhaboubi et al., 2022) or the distillation rate.

Figure 4.2 shows the instrumentation diagram of a distillation pilot plant that includes the temperature sensors, as well as the actuator scheme with the heating power control that regulates the amount of heat in the boiler.

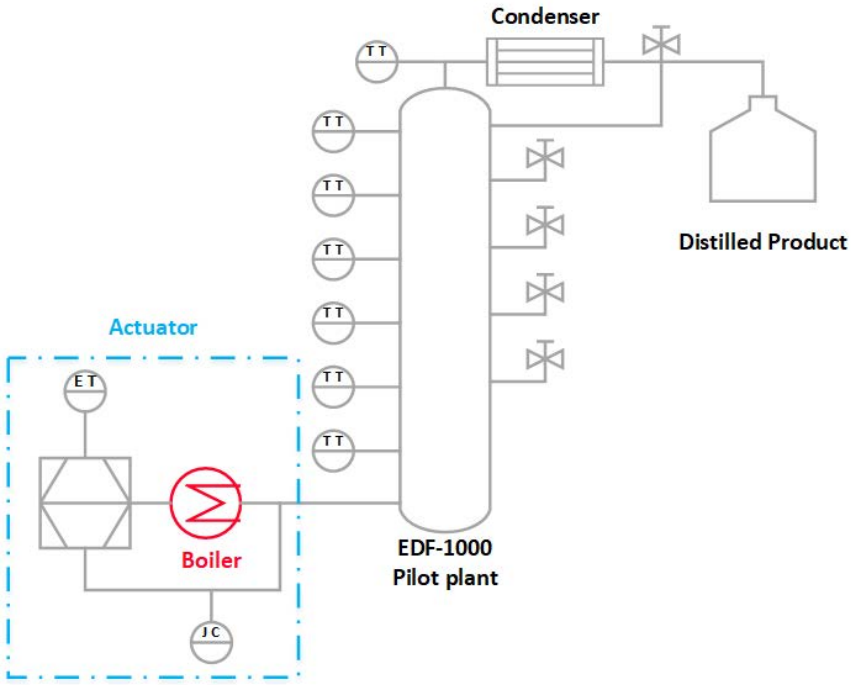


Fig. 4.2: Instrumentation diagram of the EDF-1000 distillation pilot plant. TT, TV and LP are the temperature transmitter, the voltage transmitter and power control, respectively

4.2.1 Nonlinear model of the distillation column

The distillation column model consists of a set of differential equations that represent the dynamics of each plate of the column in steady state, i.e., when the first drop is distilled. Generally, the model of a distillation column is based on the balance of the light component in the plates and is given by

$$\frac{dx_i}{dt} = \frac{V_d(g_{i+1} - g_i) + L_d(x_{i-1} - x_i)}{M_i}, \tag{4.1}$$

for $i = 2, 3, \ldots, n-1$, where V_d is the vapor molar flow, L_d the liquid molar flow, M_i the retained mass in plate i, $x_{i\pm m}$ the liquid composition in plate $i \pm m$, $g_{i\pm m}$ the vapor composition in plate $i \pm m$ with $m = 1$ and each component $x_i, g_i \in \mathbb{R} : 0 < x_i \leq 1, 0 < g_i \leq 1$.

For $i = 1$, the condenser schematic named plate 1 is shown in Fig. 4.3, and its dynamics is expressed by

$$\frac{dx_1}{dt} = \frac{V_d g_2 - L_d x_1 - D x_1}{M_1}, \tag{4.2}$$

where M_1 is the retained mass in the condenser, x_1 is the liquid composition in the condenser, g_1 is the vapor composition in the condenser, and D is the distilled product.

Thus, the body of the column is formed by $n-2$ plates. Figure 4.4 shows the schematic of a plate in the body of the column, as well as the variables that interact in the dynamics of each plate, which are expressed by (4.1).

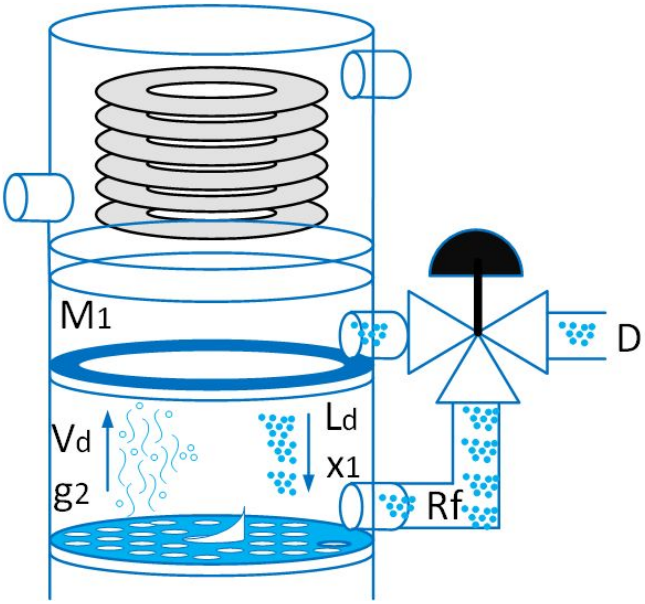


Fig. 4.3: Condenser schematic

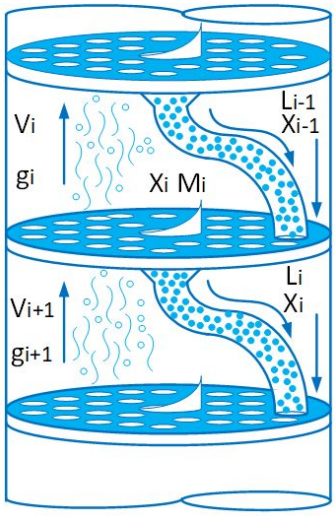


Fig. 4.4: Plate schematic

Figure 4.5 shows the schematic of the boiler in the distillation column for the plate n and its dynamics is expressed by

$$\frac{dx_n}{dt} = \frac{V_d x_n - V_d g_n + L_d x_{n-1} - L_d x_n}{M_n}, \tag{4.3}$$

where M_n is the retained mass in the boiler, x_n the liquid composition in the boiler, g_n the vapor liquid composition in the boiler, x_{n-1} the liquid composition in plate $n-1$ and n the total number of plates.

Additionally, according to Skogestad (1997), a batch type distillation column has an interaction of three molar flows:

- vapor, V_d

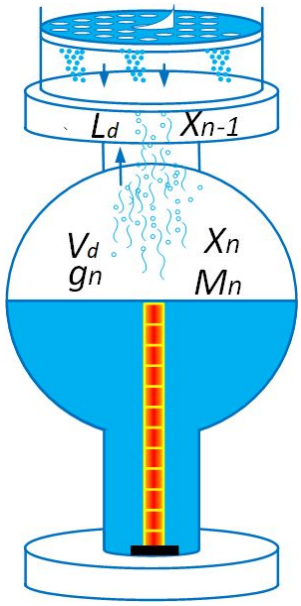


Fig. 4.5: Boiler schematic

$$V_d = \frac{Q_b}{H_i^{vap} x_n + H_j^{vap}(1 - x_n)}, \tag{4.4}$$

where Q_b is the heating power, H_i^{vap} the vapor enthalpy of the light element of the mixture and H_j^{vap} the vapor enthalpy of the heavy element of the mixture.

- liquid, L_d

$$L_d = (1 - Rf)V_d, \tag{4.5}$$

where Rf the percentage of the reflux action.

- distilled product, D

$$D = V_d - L_d. \tag{4.6}$$

In addition, the relative volatility is considered dynamic, i.e., changes over time, in the model presented in this chapter. Relative volatility is defined as the difference between the vapor pressure of the most volatile components and the vapor pressure of the less volatile components of a liquid mixture, expressed in

$$g_i P_T = P_i^{sat} x_i \gamma_i, \tag{4.7}$$

where P_i^{sat} is the saturation pressure of the mixture components, γ_i is the activity coefficient and P_T is the total pressure of the process expressed in the liquid-vapor equilibrium for nonideal mixtures. The activity coefficient γ_i is dependent on the liquid concentration of the elements in the mixtures.

4.3 Case study: EDF-1000 distillation pilot plant

The case study is the EDF-1000 distillation pilot plant shown in Fig. 4.6, consisting of 11 perforated plates, 7 of which have Pt100 RTD temperature sensors located in the condenser (plate 1), plates 2, 4, 6, 8, 10 and the boiler (plate 11).

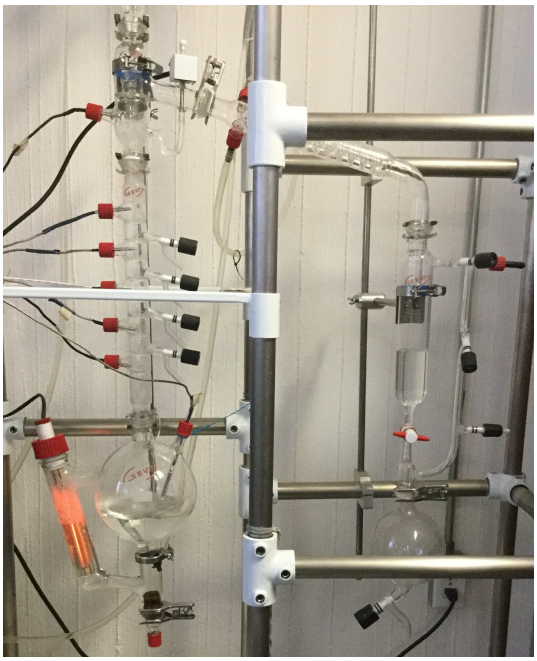


Fig. 4.6: EDF-1000 distillation pilot plant

The most important physical characteristics of the EDF-1000 distillation pilot plant are as follows:

- Two-liter boiler tank
- 350-Watts heating resistor
- Bottom product output valve
- Double spiral-condenser
- On-off reflux valve

4.3.1 State-space model of the EDF-1000 distillation pilot plant

In order to obtain the linear state-space model of the EDF-1000 distillation pilot plant with 11 plates for a binary mixture, and considering that $g_i = (1 - x_i)$, $G(x_i)$ is expressed for any operation point as

$$G(x_i) = x_i \frac{P_i^{sat} e^{A_{21}(\frac{A_{21}(1-x_i)}{A_{12}x_i + A_{21}(1-x_i)})^2}}{P_T}. \tag{4.8}$$

Thus, the model can be represented by

$$\dot{x} = \begin{pmatrix} \frac{-(V_d+D)}{M_1} & \frac{V_d \cdot G(x_1)}{M_1} & 0 & \dots & 0 \\ \frac{L_d}{M_2} & \frac{-V_d \cdot G(x_2)-L_d}{M_2} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \frac{-V_d \cdot G(x_{10})-L_d}{M_{10}} & \frac{V_d \cdot G(x_{11})}{M_{10}} \\ 0 & 0 & \dots & \frac{L_d}{M_{11}} & -\frac{L_d}{M_{11}} \end{pmatrix} x + \begin{pmatrix} \frac{V_d x_1}{M_1} & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & \frac{x_n(1-G(x_n))}{(H_{etha}^{vap} x_n + H_{H2O}^{vap}(1-x_n)M_n)} \end{pmatrix} \begin{pmatrix} R_f \\ Q_b \end{pmatrix}. \tag{4.9}$$

where the light component compositions, $x^T = [x_1, x_2, \cdots, x_{10}, x_{11}]$ are the state deviations with respect of the operation point and the heating power, Q_b, the reflux, R_f, the control inputs and A_{21} and A_{12} are the activity coefficients of the mixture components.

The system output is given by

$$y = Cx = Ix, \tag{4.10}$$

with I the identity matrix. These equations are described in detail in (Orozco et al., 2016).

The following considerations for the distillation column are assumed.

- Constant pressure throughout the column
- Inflows and outflows in the liquid state
- No vapor retention
- Vapor and liquid balance
- Vapor and liquid perfectly mixed
- Adiabatic distillation column
- Batch feeding

The simulation results of these nonlinear models are described in Orozco et al. (2016).

4.3.2 EDF-1000 heating actuator scheme and model

The boiler is the element that provides the amount of heat necessary to evaporate the mixture to be distilled. Boiler actuators generally control heating power from electricity. The amount of the generated heat output allows controlling the distillation rate in the process, according to equations (4.4) to (4.6), as shown in Fig. 4.7.

It is very important to regulate the temperature at a suitable value because in certain mixtures different temperatures represent different products, as in the case of petroleum distillation. An inadequate generation of the heat exchange between the boiler and the mixture in the distillation process can cause temperature variations, no-uniform heating in the mixture and thermal shocks in the plates, among other damage. Therefore, it is important to design FDI systems to avoid risks for the user and damage to the equipment (Paraschiv and Olteanu, 2015).

The boiler in the case study is formed by two tanks where the mixture is deposited and heated by a heating element (resistor). Figure 4.8 shows the schematic of the two-tank boiler.

The heating power is determined by Joule's Law expressed as follows: The amount of heat generated by an electric current passing through a conductor is directly proportional to the resistance of the

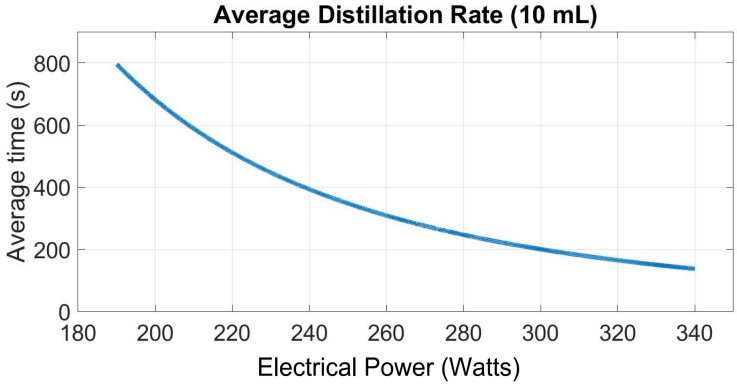


Fig. 4.7: Distillation rate versus electrical power

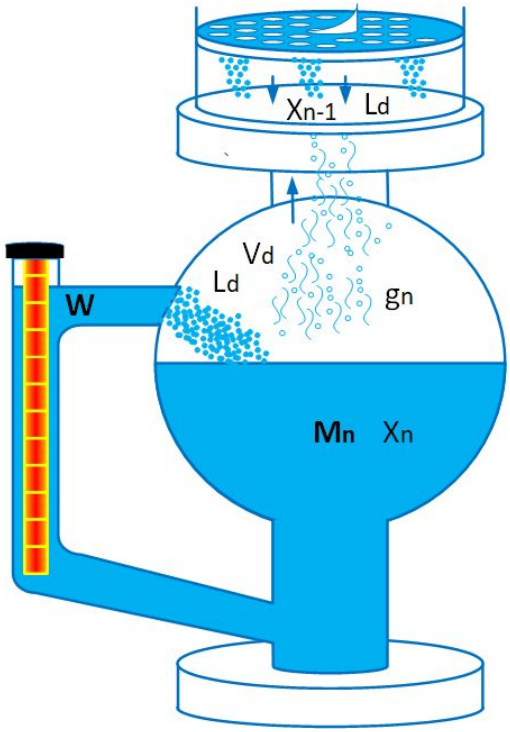


Fig. 4.8: Two-tank boiler schematic

conductor, the square of the intensity of the current and the duration of the current passing through the conductor (Li et al., 2022). Joule's law is described by

$$J = i_e{}^2 Rt, \tag{4.11}$$

where J is the heat amount (Joules), i_e the electric current (Amperes), R the resistance (Ohms) and t the time (seconds).

The law of conservation of energy states that energy cannot be created or destroyed; it can only be changed from one form to another. Joule's law expressed in electrical power W is defined by

$$J = Wt. \tag{4.12}$$

The heating resistance converts the electric energy into heat by the circulation of current. Therefore, the boiler heating power can be manipulated and modeled from the electrical power in the resistance, i.e., in the heating actuator of a distillation column, Q_b is expressed as

$$Q_b = Wt. \tag{4.13}$$

The boiler actuator scheme, shown in Fig. 4.9, adjusts the output power W_o in the boiler heating resistance by regulating the output voltage V_o with a DC-DC converter. DC-DC converters can regulate the output voltage to the desired value by switching electronic devices, usually diodes and transistors. These power electronics converters have applications in renewable energy systems, smart grids, as well as domestic and laboratory equipment power systems (Affam et al., 2021; Rojas et al., 2018).

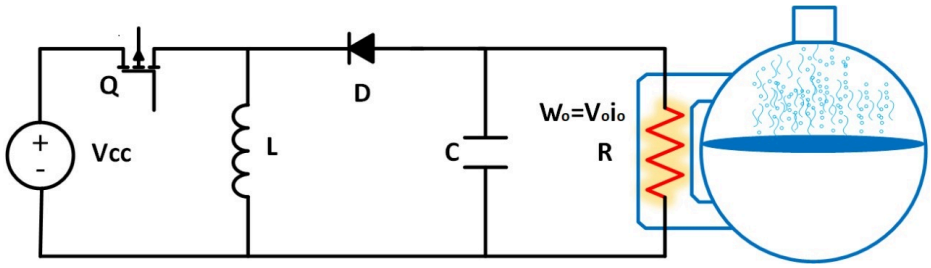


Fig. 4.9: Boiler actuator scheme using a buck-boost converter

The basic topologies of CD-CD converters are buck, boost and buck-boost (Rashid, 2017). In the buck converter, the output voltage is lower than the input voltage; in the boost converter the output voltage is greater than the input voltage.

The buck-boost converter is a combination of both converters depending on the duty cycle service d, where the output voltage V_o is given by (4.14). For d values less than 0.5, the configuration corresponds to a buck converter. On the contrary, d greater than 0.5 corresponds to a boost converter

$$V_o = -\frac{V_{cc}d}{1-d} \qquad \text{with } 0 < d < 1. \tag{4.14}$$

Input voltage, load variations, disturbances and deterioration of the power converters components (Tarakanath et al., 2014), are undesired factors that directly affect their performance, reliability and safety, hence the importance of designing and implementing FDI strategies that facilitate estimating or identifying fundamental parameters in its operation to improve safety and reliability in the system.

4.3.2.1 Buck-boost converter linear model

The linear model of the buck-boost converter is obtained from the ON - OFF states of the switching device (transistor), considering the set of equations from each topological state. The matrix representation of the model is given by

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = A \begin{pmatrix} i_L \\ v_C \end{pmatrix} + Bu, \tag{4.15}$$

where the state variables are the inductor current i_L and the capacitor voltage v_C.

During the ON-state time ($t = ON$) the converter has the topological circuit shown in Fig. 4.10, and its model is written as

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{pmatrix} \begin{pmatrix} i_L \\ v_C \end{pmatrix} + \begin{pmatrix} \frac{1}{L} \\ 0 \end{pmatrix} V_{cc}. \tag{4.16}$$

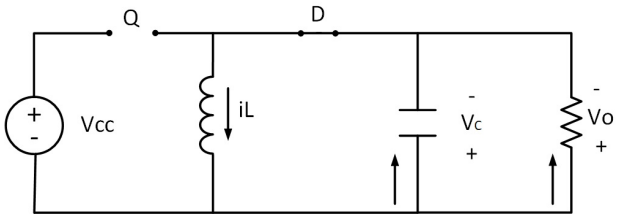


Fig. 4.10: Topological ON-state of the buck-boost converter

During the OFF-state time ($t = OFF$), the converter has the topological circuit shown in Fig. 4.11, and its model is described as

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{pmatrix} \begin{pmatrix} i_L \\ v_C \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} V_{cc}. \tag{4.17}$$

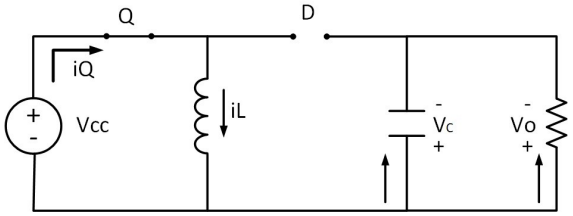


Fig. 4.11: Topological OFF-state of the buck-boost converter

The main characteristic of this model is the commutation between the two linear subsystems, represented by (4.16) and (4.17), which commute from the state of the Q switch. The general matrix representation of the system (4.15) is expressed in

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = A_k \begin{pmatrix} i_L \\ v_C \end{pmatrix} + B_k u, \tag{4.18}$$

where $k = 1,2$ characterize the subsystem for each state of the transistor. Thus,

$$A_1 = \begin{pmatrix} 0 & 0 \\ 0 & -\frac{1}{RC} \end{pmatrix},\ A_2 = \begin{pmatrix} 0 & \frac{1}{L} \\ -\frac{1}{C} & -\frac{1}{RC} \end{pmatrix},\ B_1 = \begin{pmatrix} \frac{V_s}{L} \\ 0 \end{pmatrix},\ B_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ x = \begin{pmatrix} i_L \\ v_C \end{pmatrix}.$$

4.3.2.2 Buck-boost converter nonlinear model

The nonlinear model of the converter unifies both linear subsystems and includes the control variable u, which is determined by the duty cycle $u = d$ by considering values between 0 and 1, as shown in

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = \left(A_1 \begin{pmatrix} i_L \\ v_C \end{pmatrix} + B_1 \right) d + \left(A_2 \begin{pmatrix} i_L \\ v_C \end{pmatrix} + B_2 \right) (1-d). \tag{4.19}$$

This equation can be also represented as

$$\begin{pmatrix} \dot{i}_L \\ \dot{v}_C \end{pmatrix} = A_2 \begin{pmatrix} i_L \\ v_C \end{pmatrix} + B_2 + (A_1 - A_2) \begin{pmatrix} i_L \\ v_C \end{pmatrix} d + (B_1 - B_2)d. \tag{4.20}$$

This model is considered an average model of the linear submodels. As can be observed from (4.16) to (4.20), the capacitor voltage v_C can be affected by variations in the load (heating resistor) and the input voltage, provoking faults in the actuator and, hence, in the distillation column dynamics.

4.3.2.3 Buck-boost converter Takagi-Sugeno linear model

A Takagi-Sugeno (TS) fuzzy model that interpolates between p linear submodels is based on the following model rule:

$$\begin{gathered} \textbf{Model Rule i}: \\ \textbf{if } z_1(t) \text{ is } M_{i1} \text{ and ... and } z_p(t) \text{ is } M_{ip} \\ \textbf{Then} \\ \dot{x}(t) = A_i x(t) + B_i u(t) \\ y = C_i x(t), \end{gathered} \tag{4.21}$$

where $z_i(t) \in \mathbb{R}^p$ are the fuzzy variables, M_{ip} the fuzzy sets, $x(t) \in \mathbb{R}^n$ the state vector, $u(t) \in \mathbb{R}^r$ the input vector, $y(t) \in \mathbb{R}^m$ the measurable output vectors, $C \in \mathbb{R}^{m \times n}$ an output matrix, and the matrices $A_i \in \mathbb{R}^{n \times n}$ and $B_i \in \mathbb{R}^{n \times r}$, for all i, the state and input matrices with real finite values.

Based on the nonlinear model of the buck-boost converter, presented in (4.20), and using as fuzzy variables the states ($z_1 = v_C$, $z_2 = i_L$) that operate between maximum and minimum nominal values ($z_{1max} = v_{C_{max}}$, $z_{1min} = v_{C_{min}}$, $z_{2max} = i_{L_{max}}$, $z_{2max} = i_{L_{min}}$), a Takagi-Sugeno fuzzy model that interpolates between four linear submodels based on the following rules is proposed.

According to the converter characteristics, the linear submodels are obtained using a nonlinear sector condition, where

$$A_1 = \begin{pmatrix} 0 & 1/L \\ -1/C & -1/RC \end{pmatrix} = A_2 = A_3 = A_4$$

$$B_1 = \begin{pmatrix} \frac{V_{in}+z_{1_{min}}}{L} \\ \frac{z_{2_{min}}}{C} \end{pmatrix}, \quad B_2 = \begin{pmatrix} \frac{V_{in}+z_{1_{min}}}{L} \\ \frac{z_{2_{max}}}{C} \end{pmatrix}, \quad B_3 = \begin{pmatrix} \frac{V_{in}+z_{1_{max}}}{L} \\ \frac{z_{2_{min}}}{C} \end{pmatrix}, \quad B_4 = \begin{pmatrix} \frac{V_{in}+z_{1_{max}}}{L} \\ \frac{z_{2_{max}}}{C} \end{pmatrix}$$

$$C_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = C_2 = C_3 = C_4.$$

To describe the fuzzy sets for the capacitor voltage $v_C = z_1$ the membership functions $\mu(z_1)$ are used

$$\mu_{z_{1min}}(z_1) = \begin{cases} 1 & if \ z_1 \leq z_{1min} \\ \frac{z_{1max}-z_1}{z_{1max}-z_{1min}} & if \ z_{1min} < z_1 < z_{1max} \\ 0 & if \ z_1 \geq z_{1max}; \end{cases}$$

$$\mu_{z_{1max}}(z_1) = \begin{cases} 0 & if \ z_1 \leq z_{1min} \\ 1 - \mu_{z_{1_{min}}} & if \ z_{1min} < z_1 < z_{1max} \\ 1 & if \ z_1 \geq z_{1max}; \end{cases}$$

and for the inductor current $i_L = z_2$ by

$$\mu_{z_{2min}}(z_2) = \begin{cases} 1 & if \ z_2 \leq z_{2min} \\ \frac{z_{2max}-z_2}{z_{2max}-z_{2min}} & if \ z_{2min} < z_2 < z_{2max} \\ 1 & if \ z_2 \geq z_{2max} \end{cases}$$

$$\mu_{z_{2max}}(z_2) = \begin{cases} 0 & if \ z_2 \leq z_{2min} \\ 1 - \mu_{z_{2_{min}}} & if \ z_{2min} < z_1 < z_{2max} \\ 1 & if \ z_2 \geq z_{2max}. \end{cases}$$

The normalized weights h_i are given by

$$h_1(z_1,z_2) = \mu_{z_{1min}}(z_1)\mu_{z_{2min}}(z_2), \quad h_2(z_1,z_2) = \mu_{z_{1min}}(z_1)\mu_{z_{2max}}(z_2),$$
$$h_3(z_1,z_2) = \mu_{z_{1max}}(z_1)\mu_{z_{2min}}(z_2), \quad h_4(z_1,z_2) = \mu_{z_{1max}}(z_1)\mu_{z_{2max}}(z_2).$$

Thus, the Takagi-Sugeno fuzzy model for the buck-boost converter, considering $r = 4$, A constant and $d = u(t)$ is given by

$$\dot{x}(t) = Ax(t) + \left(\sum_{i=1}^{r} h_i(z_1,z_2)B_i \right) d \tag{4.22}$$
$$y(t) = \sum_{i=1}^{r} h_i(z_1,z_2)C_i x(t).$$

4.4 Observers design

A state observer is a dynamic system that estimates state variables or parameters from available measurements. Observers, also called virtual sensors, are widely used because they estimate the system variables

that are not measurable by using mathematical algorithms and available measurements. An advantage of the observer is that it can detect and locate faults in the system. In addition, it is a systematic design procedure, which facilitates its implementation and execution in real time.

The precise mathematical model of the system to be estimated is a key point in the design since the observer recovers the behavior of the real system from its model using a closed loop scheme. (Lopez et al., 2015; Téllez-Anguiano et al., 2017; Heras-Cervantes et al., 2016). The general dynamic observer for estimating the states is

$$\dot{\hat{x}}(t) = \underbrace{A\hat{x} + Bu(t)}_{Predictor} + \underbrace{L(y(t) - \hat{y}(t))}_{Corrector}, \tag{4.23}$$

where $\hat{x}$ in $\mathbb{R}^n$ represents the state estimate for all time $\tau > t_0$ and the estimated output is given by $\hat{y}(t) = C\hat{x}(t)$.

The system presented in (4.23) is also denoted as the Luenberger identity observer and is coupled with the original process through the inputs and outputs, as shown in Fig. 4.12. The observer consists of two parts: a predictive stage, based on the model of the observed system, and a corrective stage, formed by the estimation error $e(t) = y(t) - \hat{y}(t)$, i.e., the difference between the measurable and estimated outputs.

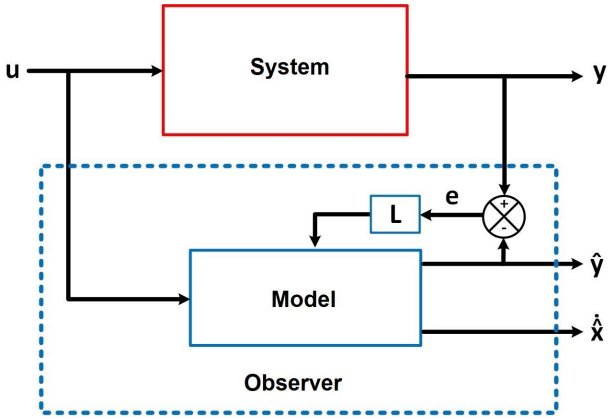


Fig. 4.12: General scheme of a state observer

4.4.1 Takagi-Sugeno fuzzy observer

Combining the Takagi-Sugeno fuzzy model of a nonlinear system with the Luenberger observer, the general structure of a fuzzy observer is obtained, according to Tanaka et al. (1998), as

$$\begin{aligned} \dot{\hat{x}}(t) &= \sum_{i=1}^{r} h_i(z(t))[A_i\hat{x}(t) + B_i u(t) + K_i e] \\ \hat{y}(t) &= \sum_{i=1}^{r} h_i(z(t))C_i\hat{x}(t). \end{aligned} \tag{4.24}$$

In Nguyen et al. (2019), the authors demonstrate the stability of the fuzzy observer as long as there is a P matrix that satisfies the linear matrix inequalities (LMIs) given by

$$\begin{gathered} P > 0 \\ N_i > 0 \\ A_i^T P - C_i^T N_i^T + P A_i - N_i C_i < 0 \\ A_i^T P - C_j^T N_i^T + P A_i - N_i C_j + P A_j^T - C_i^T N_j^T + P A_j - N_j C_i < 0, \end{gathered} \tag{4.25}$$

where P and N_i are positive definite matrices $P > 0$, $N_i > 0$ and the condition should be hold for all $i < j$. Observer gains are defined by the LMI's system solution defined in

$$K_i = P_o^{-1} N_i. \tag{4.26}$$

4.4.2 Sliding-Mode fuzzy observer

The sliding-mode fuzzy observer is based on the Luenberger observer for linear systems and the fuzzy observer proposed by Tanaka et al. (1998). By using a fuzzy observer, sliding-mode local observers can be built for each linear subsystem. Each observer is associated with a fuzzy rule i defined by

$$\begin{gathered} \textbf{Fuzzy Rule i}: \\ \textbf{if } z_1(t) \text{ is } M_{i1} \text{ and ... and } z_p(t) \text{ is } M_{ip} \\ \textbf{Then} \\ \dot{\hat{x}}(t) = A_i \hat{x}(t) + B_i u(t) + K_i e + \varphi_i(t) \\ \hat{y} = C_i \hat{x}(t). \end{gathered} \tag{4.27}$$

The final observer is given by the weighted sum of each subsystem, as shown in

$$\begin{gathered} \dot{\hat{x}}(t) = \sum_{i=1}^{r} h_i(z(t))(A_i \hat{x}(t) + B_i u(t) + K_i(e) + \varphi_i(t)) \\ \hat{y}(t) = \sum_{i=1}^{r} h_i(z(t))(C \hat{x}(t)). \end{gathered} \tag{4.28}$$

The term $\varphi_i(t)$ is the discontinuous vector of sliding modes for the subsystem i, defined by

$$\varphi_i(t) = E_{fi} sign(P_i \tilde{e}(t)), \tag{4.29}$$

where the sign function of $P_i \tilde{e}(t)$ is calculated element by element, $E_{fi} > 0$ a positive constant, $P_i > 0$ satisfies the Lyapunov equation, and the estimated state error $\tilde{e}$ is defined by

$$\tilde{e} = x(t) - \hat{x}(t). \tag{4.30}$$

4.4.3 Sliding-Mode Takagi-Sugeno fuzzy observer

From the Takagi-Sugeno observer presented in (4.24), according to Castillo et al. (2005), the corresponding sliding-mode Takagi-Sugeno fuzzy observer is defined by

$$\begin{aligned} \dot{\hat{x}}(t) &= \sum_{i=1}^{r} h_i(z(t))(A_i\hat{x}(t) + B_i u(t) + K_i(e) + \varphi_{TSi}(t)) \\ \hat{y}(t) &= \sum_{i=1}^{r} h_i(z(t))(C\hat{x}(t)). \end{aligned} \tag{4.31}$$

with the sliding vector $\varphi_{TSi}(t)$ is defined as

$$\varphi_{TSi}^{T}(t) = sign(\dot{\tilde{e}}^{T} P_i). \tag{4.32}$$

is considered as the product between the estimation error derivative $\dot{\tilde{e}}$ and a positive definite matrix P_i, where the dynamic estimation error corresponds to the difference between the dynamic measured states of the system and the dynamic states estimated by the sliding-mode fuzzy observer, as expressed in

$$\dot{\tilde{e}} = \dot{x}(t) - \dot{\hat{x}}(t). \tag{4.33}$$

Since we are interested in the error approaching zero as t approaches infinity. According to the analysis and design characteristics between the fuzzy observer and the sliding-mode fuzzy observer, the stability and the matrix P are determined as in (4.25) and the gains K_i as in (4.26).

4.5 Design of the sliding-mode Takagi-Sugeno fuzzy observer for the boiler heating actuator

As mentioned, the buck-boost converter is the boiler heating actuator. According to the Takagi-Sugeno fuzzy model for the buck-boost converter defined in (4.22), the corresponding fuzzy observer is defined by

$$\begin{aligned} \dot{\hat{x}} &= Ax(t) + \left(\sum_{i=1}^{4} h_i(z_1, z_2) B_i + \varphi_{TSi}(t) \right) d \\ y(t) &= \sum_{i=1}^{4} h_i(z_1, z_2) C x(t). \end{aligned} \tag{4.34}$$

The output matrix C is defined by

$$C_{=} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{4.35}$$

where the system outputs are v_C and i_L.

The block diagram of the buck-boost converter observer is shown in Fig. 4.13, where the fuzzy variables (z_1, z_2) are the states of the system ($x_1 = v_C$, $x_2 = i_L$), and the gains of the fuzzy observer are defined as K_f and K_ϕ.

According to the characteristics of the fuzzy model of the converter presented in (4.22), where the state matrices A_1, A_2, A_3 and A_4 are identical, the LMI that guarantees the stability of the sliding-mode

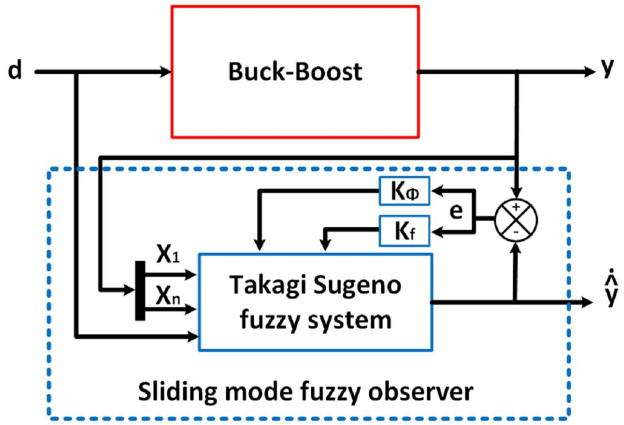


Fig. 4.13: Sliding-mode Takagi-Sugeno fuzzy observer for the buck-boost converter

fuzzy observer for the buck-boost converter is defined by

$$\begin{gathered} P_{\phi_a}\phi > 0 \\ A'P_{\phi_a} - C'N'_{\phi_a} + P_{\phi_a}A - N_{\phi_a}C < 0. \end{gathered} \tag{4.36}$$

Given the solution for P_{ϕ_a}, the gain K_ϕ for the observer is determined by

$$K_\phi = P_{\phi_a}^{-1} N_{\phi_a}. \tag{4.37}$$

4.5.1 Validation of the sliding-mode Takagi-Sugeno fuzzy observer for the heating actuator

The buck-boost converter (boiler heating actuator) observer is validated in simulation considering the converter characteristics presented in Table 4.1. Figure 4.14 shows the observer convergence to the capacitor voltage v_C under disturbances in the nominal input voltage ($V_{in} = 180$ V). In the observer simulation at 0.1 s the input voltage V_{in} decreases 8.88% ($V_{in} = 160$ V) of its nominal value. At 0.2 s it increases 111.11% ($V_{in} = 200$ V) of its nominal value. In both cases it is shown that the observer ($\hat{v_C}$) converges to the capacitor voltage v_C measured in the real system. The observer presents a maximum estimation error of 1.3 V and a minimum of 100 μ V.

Figure 4.15 shows the observer convergence to the inductor current i_L under disturbances in the nominal input voltage ($V_{in} = 180V$). In 0.1 s the voltage V_{in} decreases 88.88% to $V_{in} = 160$ V, and in 0.2 s it increases 111.11% $V_{in} = 200$ V. In both cases, it is shown that the observer $\hat{i}_L$ converges to the inductor current i_L of the nonlinear model. The observer presents a maximum estimation error of 394 mA and a minimum error of 80 μA.

Figure 4.16 shows the observer's convergence to the capacitor voltage under variations in the nominal load ($R_L = 70.3\ \Omega$). At 0.1 s the load decreases 78.23% ($R_L = 55\ \Omega$), and at 0.2 s it increases 113.79%

Table 4.1: Parameter of the buck-boost converter

Parameter	Magnitude
Input voltage (V_{cc})	$180V$
Output voltage (V_{out})	$-229V$
Inductor (L)	$5\mu H$
Capacitor (C)	$78\mu F$
Load (R_L)	70.3Ω
Frequency (f)	20 kHz
Duty cycle (d)	0.56

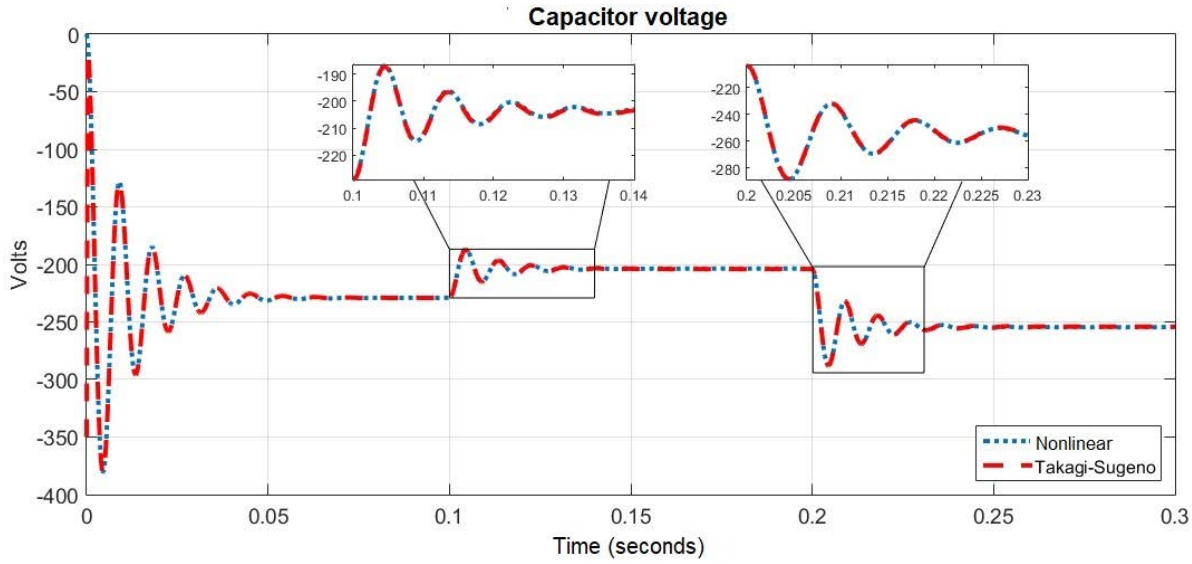


Fig. 4.14: Observer response in the capacitor voltage v_C under disturbances in the input voltage V_{in}

($R_L = 80\ \Omega$). In both cases the observer ($\hat{\dot{i}}_L$) converges to the capacitor voltage v_C of the nonlinear model. The observer has a maximum error of 1 V and a minimum error of 75 μV.

Figure 4.17 shows the observer's convergence to the inductor current i_L under variations in the nominal magnitude of the load ($R_L = 70.3\ \Omega$). At 0.1 s the load decreases 78.23% ($R_L = 55\ \Omega$), and at 0.2 s it increases 113.79% ($R_L = 80\ \Omega$). In both cases, the observer converges to the inductor current. The observer presents a maximum error of 13 mA and a minimum error of 100 μA.

As can be observed, the estimated states by the fuzzy observer for the boiler actuator converges adequately to the real states of the system under perturbations, allowing designing and implementing a fault detection and diagnosis systems based on analytical redundancy.

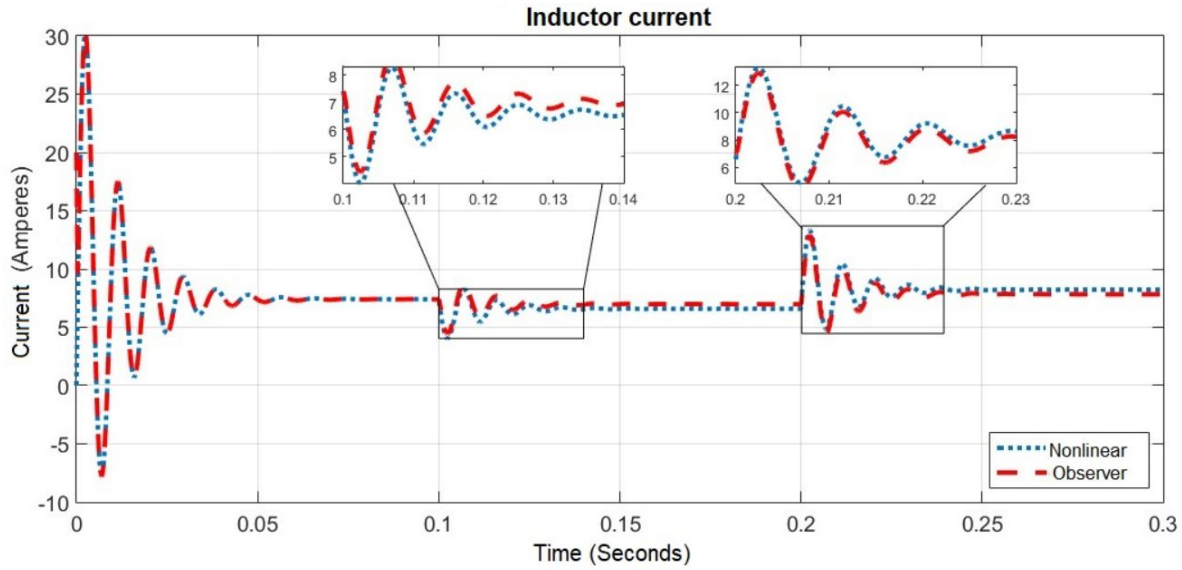


Fig. 4.15: Observer response for the inductor current i_L under disturbances in the input voltage V_{in}

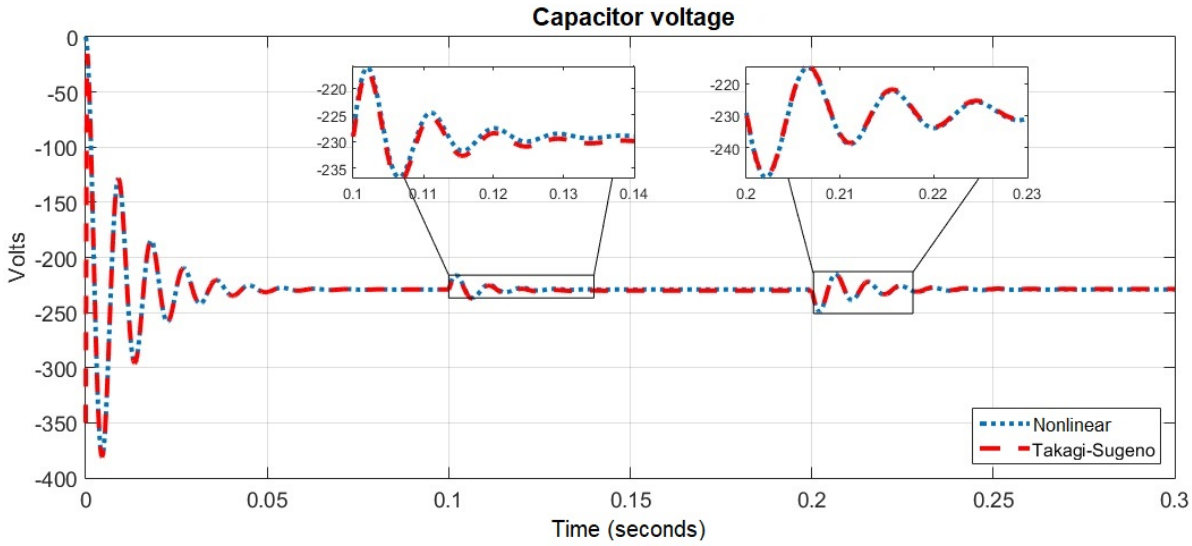


Fig. 4.16: Observer response for the capacitor voltage v_C under load variations R

4.5.2 Fault detection for the boiler heating actuator

The block diagram of the FDI system for the boiler heating actuator for the distillation column is shown in Fig. 4.18.

The FDI system for the heating actuator is based on a bank of two fuzzy observers. The inputs are the inductor current i_L for Observer 1 and the capacitor voltage v_C for Observer 2. The difference between the estimates of both observers allows detecting and diagnosing the type of fault in the converter. The difference between the estimates of both observers allows detecting and diagnosing the faults in the converter. The four generated residuals are

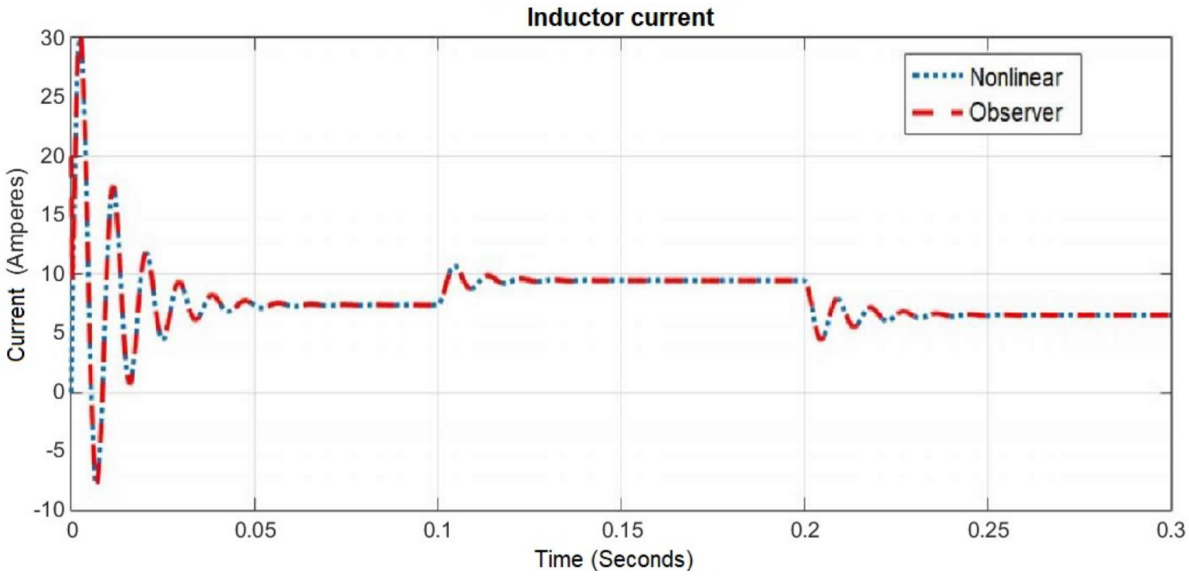


Fig. 4.17: Observer response in the inductor current i_L under load variations R

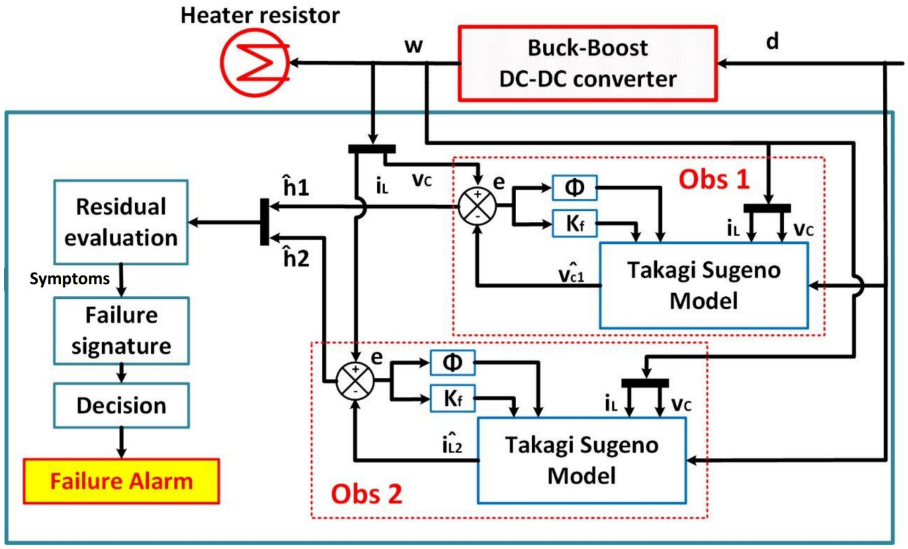


Fig. 4.18: FDI system for the boiler heating actuator

$$\begin{aligned} r_{i_L}1 &= i_L - \hat{i}_{LO1}, \quad r_{v_C}1 = v_C - \hat{v}_{CO1} \\ r_{i_L}2 &= i_L - \hat{i}_{LO2}, \quad r_{v_C}2 = v_C - \hat{v}_{CO2}, \end{aligned} \tag{4.38}$$

where $\hat{i}_{L0i}$ and $\hat{v}_{CL0i}$ for $i = 1,2$ are the outputs estimated by the observers. The residuals evaluation provides the symptoms for the diagnosis of the faults in the converter according to the fault signature presented in Table 4.2.

The FDI system detects faults in the nominal load R_L. Furthermore, it is sensitive to variations in the nominal supply voltage V_{cc}, detecting the conditions and faults below.

Table 4.2: Fault signature

Fault	F1	F2	F3	F4	F5	F6	F7	F8
r_{i_L1}	-1	0	1	-1	1	-1	1	-1
r_{i_L2}	0	0	1	-1	1	- 1	1	0
r_{v_C1}	1	-1	1	-1	1	-1	1	-1
r_{v_C2}	1	-1	0	0	1	-1	-1	1

- F1: Decrease in the nominal supply voltage of the converter V_{cc}
- F2: Increase in the nominal supply voltage of the converter V_{cc}
- F3: Decrease in the magnitude of the nominal load R_L
- F4: Increase in the magnitude of the nominal load R_L
- F5: Decrease in the nominal power supply voltage of the converter V_{cc} and decrease in the magnitude of the nominal load R_L
- F6: Increase in the nominal power supply voltage of the converter V_{cc} and increase in the magnitude of the nominal load R_L
- F7: Increase in the nominal power supply voltage of the converter V_{cc} and decrease in the magnitude of the nominal load R_L
- F8: Decrease in the nominal power supply voltage of the converter V_{cc} and increase in the magnitude of the nominal load R_L

4.6 Experimental validation of the FDI system

The experimental validation of the FDI system is performed for a buck-boost converter that regulates the voltage to a heating resistance of 350 W for the EDF-1000 distillation column boiler. The observer design parameters for the FDI system are determined for the case study presented in Table 4.1.

The FDI system is designed assuming that variations in the converter input voltage are caused mainly by the input voltage (line voltage) resulting in thermal shocks if the power supplied to the boiler increases or the slowness of the process dynamics if the power supplied in the heating resistance is low. Load variations are usually caused by degradation or manufacturing of the heating resistance.

The structure of the observer system for the buck-boost converter is expressed by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + \left(\sum_{i=1}^{4} h_i(z_1, z_2) B_i \right) d(t) \\ y(t) &= \sum_{i=1}^{4} h_i(z_1, z_2) C_i x(t). \end{aligned} \tag{4.39}$$

According to the characteristics of the fuzzy system, where $A_1, A_2, A_3, A_4 = A$, the LMIs to determine the stability of the system are given by

$$\begin{aligned} P &> 0 \\ A^T P - C^T N^T + PA - NC &< 0. \end{aligned} \tag{4.40}$$

Thus, given the solution for P, the observer gain K is determined by

$$K = P^{-1}N. \tag{4.41}$$

The FDI experimental validation for the process presented in Table 4.1 considers faults in the nominal load value, variations in the converter input voltage and the combination of both simultaneous faults.

4.6.1 Test 1: Load decrease

In order to validate the FDI behavior under heating resistance variations, a decrease of 50% of its nominal value (38 Ω) occurs at 0.1s.

Figure 4.19 presents the observers' dynamics when the load decrease fault occurs. The observer with the i_L reference, Observer 1, presents a greater difference concerning the capacitor voltage value when a load decrease occurs, causing a residue of greater magnitude with respect to the observer with the v_C reference, Observer 2.

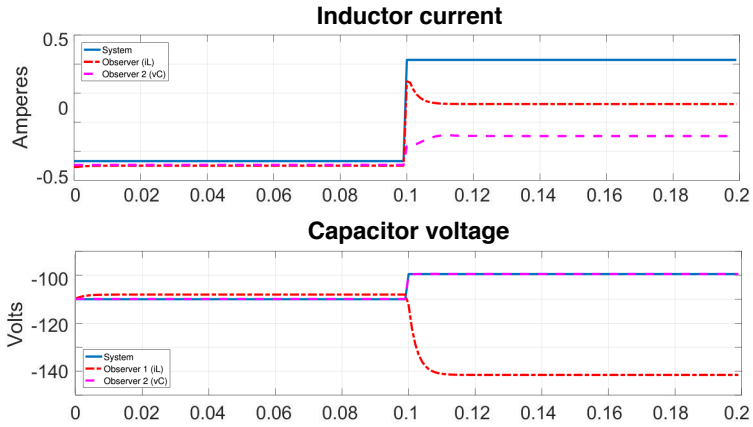


Fig. 4.19: Observers' estimation concerning a nominal load decrease

Figure 4.20 shows the residuals generated without fault from 0 to 0.1 s and under a fault's presence after 0.1 s. It can be observed that the residuals generated by Observer 1, with the i_L reference, exceed the decision thresholds, fixed experimentally according to the process dynamics, after the load decrease.

Figure 4.21 shows the symptoms generated when evaluating the residuals with the defined thresholds. According to the Table 4.2, the FDI system adequately identifies the F3 fault (nominal load decrease).

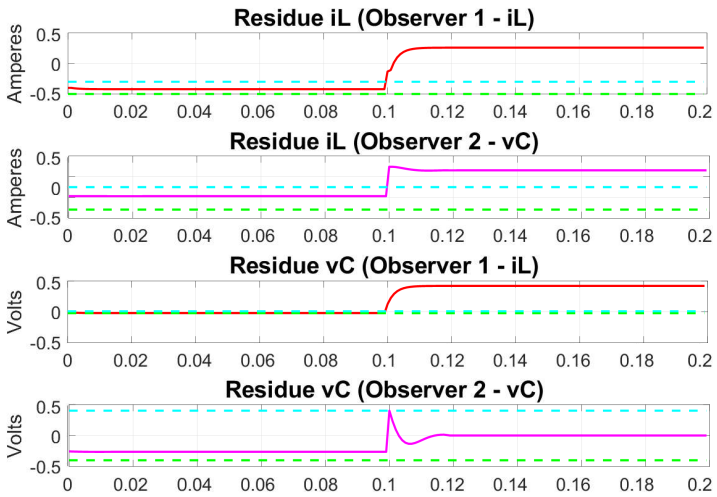


Fig. 4.20: Residuals compared to the fixed decision thresholds under a nominal load decrease

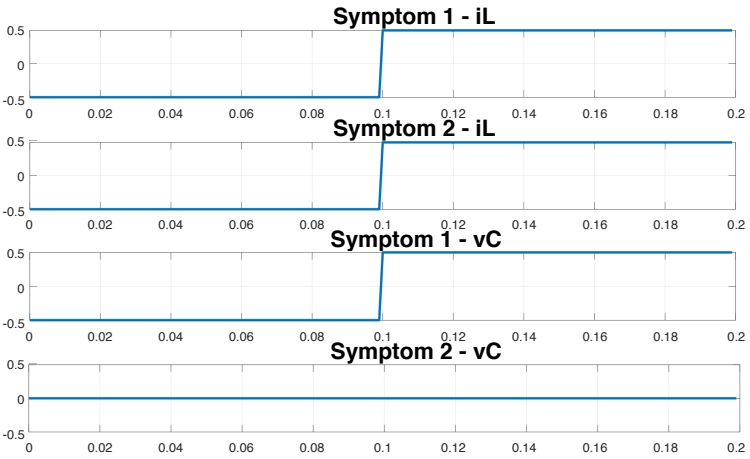


Fig. 4.21: Symptoms obtained under a nominal load decrease

4.6.2 Test 2: Input voltage decrease

In the second validation test, a disturbance in the input voltage is considered. A 16.66% decrease from its nominal value is generated at 0.1 s. Figure 4.22 shows that Observer 1 has a greater difference considering the inductor current and the capacitor voltage that Observer 2.

Figure 4.23 shows the residuals generated without fault from 0 to 0.1 s and under an input voltage decrease after 0.1 s. It can be observed that only the residuals generated by Observer 2 for the inductor current are minor than the decision thresholds fixed experimentally according to the process dynamics.

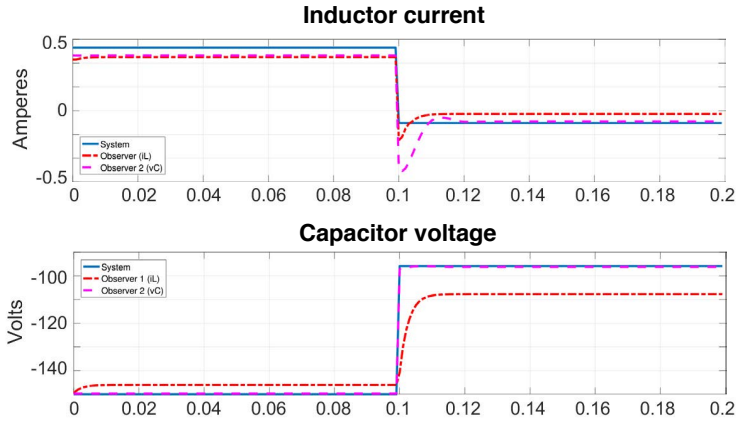


Fig. 4.22: Estimation using observer 1 and 2 considering an input voltage decrease

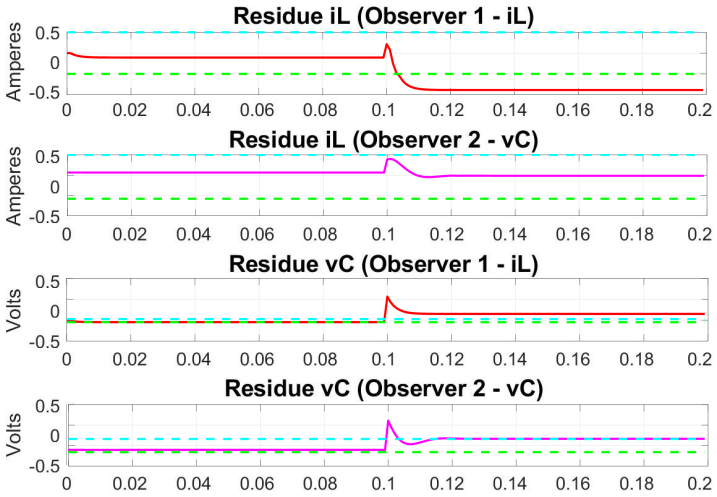


Fig. 4.23: Residuals compared to the fixed decision thresholds under an input voltage decrease

Figure 4.24 shows the symptoms generated when evaluating the residuals with the defined thresholds, and it can be observed that at 0.1 s the magnitude of the symptoms changes according to the difference between the observers estimation with respect to the real value. Symptom 2 for the v_C obtained by Observer 2 has a transient response that switches between 0 and 1 because the response of Observer 2 for the inductor current i_L presents an overdamped transient response. By evaluating the symptoms and according to Table 4.2 it can be determined that a fault exists, which corresponds to a decrease in the input voltage.

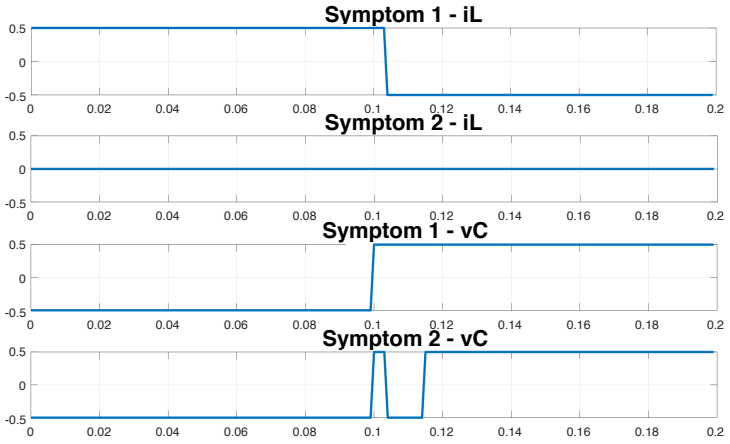


Fig. 4.24: Symptoms obtained under an input voltage decrease

4.6.3 Test 3: Simultaneous faults

A third experiment is performed where a 50% increase in the nominal value of the load in addition to a 16.6% increase in the actuator input voltage. Figure 4.25 shows the dynamics of the observers.

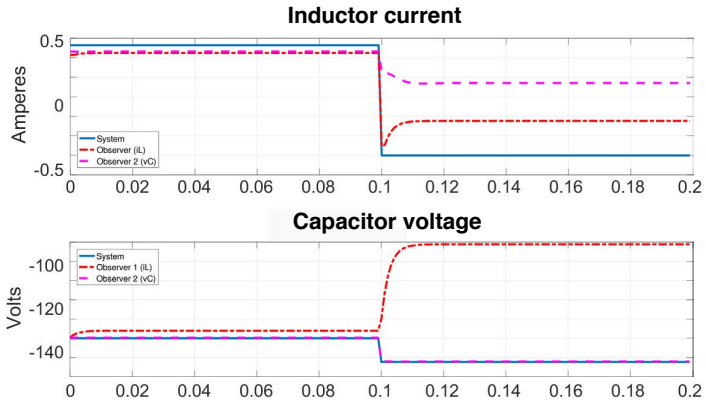


Fig. 4.25: Observers' estimation concerning under an input voltage and load decrease

At 0.1s the combined fault is presented. Both observers lose convergence with the inductor current, i_L. Observer 2 maintains convergence with the capacitor voltage despite the multiple faults. Figure 4.26 shows the residuals obtained for each observer estimation under multiple faults in the system.

The generated residuals show similar behavior with difference values below the selected thresholds. Accordingly, the symptoms that appear in Fig. 4.27 are all negative. By comparing these signatures with

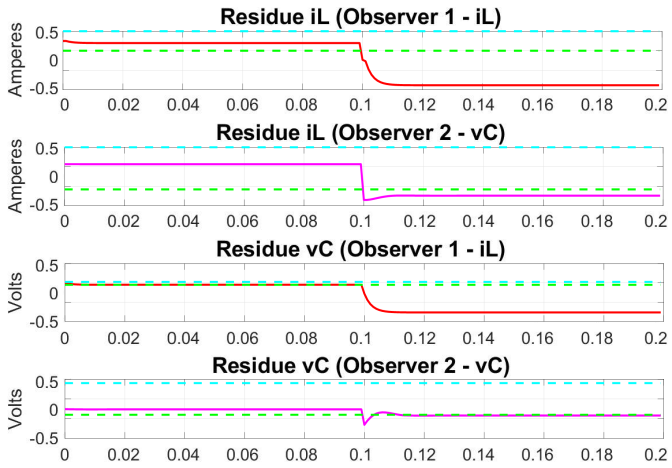


Fig. 4.26: Residuals obtained under an input voltage and load decrease

the fault shown in Table 4.2, it can be concluded that a multiple fault corresponding to the combination of load and input voltage increases exists.

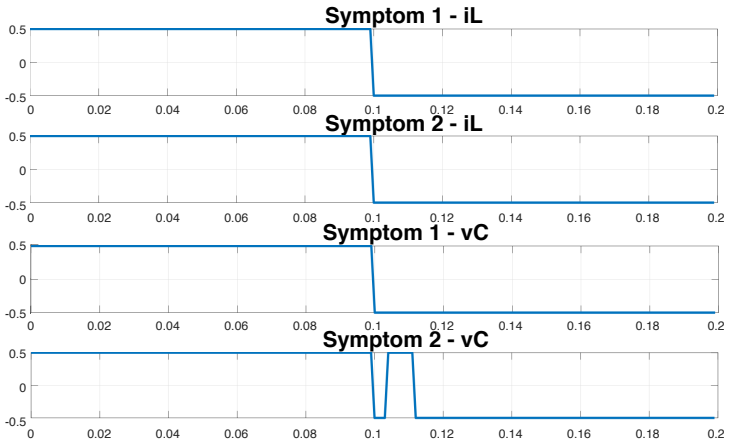


Fig. 4.27: Symptoms obtained under simultaneous input voltage and load decrease

4.7 Conclusions

In this chapter, an FDI system for a buck-boost converter acting as the boiler heating actuator for the EDF-1000 distillation pilot plant was implemented. The buck-boost converter function is to regulate the electrical power W of a heating resistance through its duty cycle d. For the FDI system, two sliding-mode

Takagi-Sugeno fuzzy observers were designed, both observers estimate the converter output voltage v_C and the inductor current, i_L.

This FDI system allows obtaining four residuals, $r_{i_L}1$, $r_{i_L}2$, $r_{v_C}1$ and $r_{v_C}2$, which determine the symptoms to indicate the presence or absence of faults.

According to the results obtained in simulation and in experimentation, the observers with sliding modes have acceptable results with small convergence times, around 800 μs.

It is feasible to experimentally design an FDI scheme for an EDF-100 distillation pilot plant modeled by a Takagi-Sugeno type structure and techniques based on the model, such as the use of observers with sliding modes.

This page intentionally left blank

References

Kumar, A., Bhattacharya, A. and Flores-Cerrillo J.F. (2020). Data-driven process monitoring and fault analysis of reformer units in hydrogen plants: Industrial application and perspectives. In *Computers & Chemical Engineering*. Volume 136, 106756. Elsevier

Affam, A., Buswig, Y. M., Othman, A. K. B. H., Julai, N. B., & Qays, O. (2021). A review of multiple input DC-DC converter topologies linked with hybrid electric vehicles and renewable energy systems. In *Renewable and Sustainable Energy Reviews*, Volume 135, 110186.

Bahreini, M., Zarei, J., Razavi-Far, R., and Saif, M. (2021). Robust and reliable output feedback control for uncertain networked control systems against actuator faults. In *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(4), 2555-2564.

Téllez-Anguiano, A. C., Heras-Cervantes, M., Anzurez-Marín, J., Chávez-Campos, G. M., and Gnecchi, J. A. G. (2017). Mathematical modelling of batch distillation columns: A comparative analysis of non-linear and fuzzy models. IntechOpen.

Demidova, G., Rassõlkin, A., Vaimann, T., Kallaste, A., Zakis, J. and Suzdalenko, A. (2021). An Overview of Fuzzy Logic Approaches for Fault Diagnosis in Energy Conversion Devices. In *2021 28th International Workshop on Electric Drives: Improving Reliability of Electric Drives (IWED)*. pages 1–7, IEEE.

Heras-Cervantes, M., Anzurez-Marin, J., del Carmen Téllez-Anguiano, A., del Carmen García-Ramírez, M., and Correa-Gómez, J. (2016). Modelling a heating-power actuator for a distillation column boiler. pages 1–6. IEEE.

Li, Y., Sun, Y., Gao, Y., Sun, J., Lyu, H. F., Yu, T., and Wang, Y. (2022). Analysis of overload induced arc formation and beads characteristics in a residential electrical cable. *Fire Safety Journal*, 131, 103626.

Ibrahim, D., Jobson, M., Li, J., and Guillén-Gosálbez, G. (2018). Optimization-based design of crude oil distillation units using surrogate column models and a support vector machine. *Chemical Engineering Research and Design*, 134:212–225.

Iqbal, R., Maniak, T., Doctor, F. and Karyotis C. (2019). Fault Detection and Isolation in Industrial Processes Using Deep Learning Approaches. *IEEE Transactions on Industrial Informatics*, volume 15, 5:3077-3084.

Kordestani, M., Saif, M., Orchard, M. E., Razavi-Far, R. and Khorasani, K. (2019). Failure prognosis and applications—A survey of recent literature. In *IEEE transactions on reliability*, volume 70, number 2, pages 728–748. IEEE.

Khan, A. S., Khan, A. Q., Iqbal, N., Sarwar, M., Mahmood, A., and Shoaib, M. A. (2020). Distributed fault detection and isolation in second order networked systems in a cyber–physical environment. *ISA transactions*, 103:131–142.

Lopez, F.-R., Ponsart, J.-C., Theilliol, D., Astorga-Zaragoza, C., and Flores-Montiel, M. (2015). Robust state and fault estimation observer for discrete-time d-lpv systems with unmeasurable gain scheduling functions. application to a binary distillation column. *IFAC-PapersOnLine*, 48(21):1012–1017.

Ming, L. and Zhao, J. (2017). Review on chemical process fault detection and diagnosis. In *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pages 457–462. IEEE.

Orozco, G., Cortés, B., Heras, M., Téllez, A., and Anzurez, J. (2016). Analysis and comparison of distillation column models considering constant and variable relative volatility. pages 1–6. IEEE.

Paraschiv, N. and Olteanu, M. (2015). Feedforward process control of a distillation column based on evolutionary techniques. In *System Theory, Control and Computing (ICSTCC), 2015 19th International Conference on*, pages 730–735. IEEE.

Rashid, M. H. (2017). *Power electronics handbook*. Butterworth-Heinemann.

Rojas, C. A., Kouro, S., Perez, M. A., and Echeverria, J. (2018). Dc-dc mmc for hvdc grid interface of utility-scale photovoltaic conversion systems. *IEEE Transactions on Industrial Electronics*, 65(1):352–362.

Alhaboubi, N. A. (2022). Constant distillate composition of batch distillation column with variable reflux mode based on still pot concentration. In *Science and Technology for Energy Transition*, 77, 2.

Skogestad, S. (1997). Dynamics and control of distillation columns: A tutorial introduction. *Chemical Engineering Research and Design*, 75(6):539–562.

Tanaka, K., Ikeda, T., and Wang, H. O. (1998). Fuzzy regulators and fuzzy observers: relaxed stability conditions and lmi-based designs. *IEEE Transactions on fuzzy systems*, 6(2):250–265.

Nguyen, A. T., Taniguchi, T., Eciolaza, L., Campos, V., Palhares, R., and Sugeno, M. (2019). Fuzzy control systems: Past, present and future. *IEEE Computational Intelligence Magazine*, 14(1), 56-68.

Tarakanath, K., Patwardhan, S., and Agarwal, V. (2014). Internal model control of dc-dc boost converter exhibiting non-minimum phase behavior. pages 1–7. IEEE.

Téllez, A. (2010). *Supervisión Electrónica de Columnas de Destilación Basada en Modelos*. PhD thesis, Centro Nacional de Investigación y Desarrollo Tecnológico, Departamento de Ingeniería Mecatrónica.

Verde, C., Gentil, S., and Morales-Menéndez, R. (2013). *Monitoreo y diagnóstico automático de fallas en sistemas dinámicos*. Trillas.

Zhong, W., Lu, J., and Miao, Y. (2017). Fault detection observer design for fractional-order systems. In *2017 29th Chinese Control And Decision Conference (CCDC)*, pages 2796–2801. IEEE.

Castillo-Toledo, B., and Anzurez-Marin, J. (2005, June). Model-based fault diagnosis using sliding mode observers to Takagi-Sugeno fuzzy model. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control*, pp. 652-657. IEEE.

Chaouech, L., Soltani, M., and Chaari, A. (2019). Fuzzy sliding mode controller design using scalar sign function for a class of TS fuzzy models. In *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 1823-1828. IEEE.

Chapter 5

Adaptive multivariate degradation model for remaining useful life prediction

Adriana Villalón Falcón[1], Alberto Prieto Moreno[1], Marcos Quiñones-Grueiro[2], and Orestes Llanes-Santiago[1]

Abstract Accuracy in predicting the remaining useful life (RUL) of industrial systems is crucial for maintenance tasks. Deep learning-based methods are among the most widely used for RUL prediction because of their ability to learn complex non-linear relationships between system measurements and RUL. After training, however, these models are incapable of adapting to systems with dynamic behavior that changes over time. On the other hand, prognostic methods based on degradation models, while capable of adapting to these changes, are usually designed to model a single variable. In this chapter, an adaptive method for predicting RUL based on modeling the behavior of multiple variables during degradation is proposed. The information from each model is combined to predict the RUL of the system. A new performance metric is proposed to evaluate the prediction models by periodically calculating the prediction error and establishing a direct relationship between RUL prediction and maintenance planning. The proposed method is applied to the NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data set, and demonstrates satisfactory results.

Nomenclature

$\alpha_{RUL_{i,t}}$	weight of RUL (remaining useful life) prediction error of system i at time t
CDF^{-1}	inverse cumulative distribution function
EOL_i	true EOL (end of life) of system i
$EOL^*_{j,i,t}$	predicted EOL with feature j of system i at time t
ET_j	EOL threshold of feature j
f_j	degradation model of feature j
N	number of systems
$\mathbf{p}$	vector of parameters of f
$\mathbf{p}_L, \mathbf{p}_U$	lower and upper bounds of parameters $\mathbf{p}$
R_s	constrained region of the parameter space
$RUL_{i,t}$	true RUL of system i at time t

[1]Universidad Tecnológica de La Habana José Antonio Echeverría, CUJAE, La Habana, Cuba,
e-mail: avillalon@automatica.cujae.edu.cu, albprieto@automatica.cujae.edu.cu and orestes@tesla.cujae.edu.cu
[2]Institute for Software Integrated Systems, Vanderbilt University, Nashville, Tennessee, USA, e-mail: marcos.quinones.grueiro@vanderbilt.edu

$RUL^*_{i,t}$	predicted RUL of system i at time t
$RUL^*_{j,i,t}$	predicted RUL with feature j of system i at time t
RUL_w	value of RUL from which the precision of the RUL prediction is considered critical
$RWMSE$	global prediction error of a set of systems
$RWMSE_i$	global prediction error of system i
t_d	detection time of the degradation process
T	number of observations
V	number of features
w_j	weight of feature j
y_j	feature j

5.1 Introduction

Deciding the amount of time that industrial facilities can continue operating and maintaining quantitative and qualitative production standards is generally based on corrective or preventive maintenance strategies. Both maintenance approaches considerably increase production costs and product delivery times. Predictive maintenance, on the other hand, aims to minimize the downtime of systems and maximize their useful life by predicting when a system cannot operate satisfactorily anymore (Niu, 2017; Kim et al., 2016; Lei et al., 2018).

The planning of the predictive maintenance tasks and their correct execution to a large extent depends on the prediction of the RUL. Because of the large amount of data provided by data acquisition systems, many methods capable of providing accurate predictions of RUL from measurements of process variables have been developed.

Among the most widely used data-driven methods are deep neural networks (DNNs), because of their ability to model complex nonlinear relationships (Ellefsen et al., 2018; Guo et al., 2017; Li et al., 2018; Yang et al., 2019; Zheng et al., 2017; Wu et al., 2018). Some network architectures have been designed to consider temporal features that reflect the degradation of systems (Ellefsen et al., 2018; Zheng et al., 2017; Wu et al., 2018). Configuring a DNN takes time, however, and is computationally expensive. After training, a fixed model is obtained that cannot adapt to the system changing conditions over time. If system degradation exhibits new behavior, the DNN must be trained again.

Methods based on degradation models predict the RUL by adapting to the systems' conditions during the degradation process (Li et al., 2015; Feng et al., 2016; Lei et al., 2016; Ali et al., 2018; Saidi et al., 2017; Yan et al., 2020). These methods only model one variable. The prediction of the RUL, however, is not determined by one variable but by the simultaneous evolution of multiple variables. Therefore, multivariable analysis is required. This chapter proposes an adaptive method for predicting the RUL based on modeling the behavior of features that provide significant information during the degradation process. The information from each model is considered for predicting the RUL of the system.

An important element in the configuration of prognostic systems is the selection of the adequate models to predict the RUL because of the constant development of new prediction methods designed for different applications. Although sometimes the selection of a method is based on a particular application, often it is necessary to compare several models according to their performance in the prediction

task. Therefore, performance metrics are used to assess the RUL prediction error. The most commonly used performance metrics are related to the accuracy and precision of prognostic methods (Saxena et al., 2008; Lei et al., 2018; Zeng et al., 2017). This chapter will focus on accuracy metrics, given their importance in prognostic performance. A wide range of metrics have been developed to measure the accuracy of prognostic methods. Accuracy metrics quantify the similarity between the model prediction and true measured values (Zeng et al., 2017). Generally, metrics compute the similarity as the difference between the RUL predicted values (RUL^*) and true values (RUL), Δ, also called prediction error (5.1):

$$\Delta = RUL^* - RUL. \tag{5.1}$$

Accuracy metrics are created by modifying (5.1) to add desired features for metrics based on the capability of prognostic methods for supporting maintenance decisions. The most relevant features that metrics include, ordered by their importance, are detailed below:

1. *Overall performance*. Accuracy must be measured over the lifetime of a system, capturing the prediction error behavior. Some metrics have been used to measure the prediction error in an instant of the system degradation. RUL prediction is a continuous process; thus, the evaluation of methods for this purpose requires measuring how the prediction error changes over time. It is not enough to consider the error at specific time instants as a unique evaluation measure.
2. *Metric value in time units*. Accuracy values must be defined in time units according to the RUL measurement (i.e., hour, day, cycle). The time unit of the prediction is vital for measuring the accuracy of prognostic methods, which allows establishing a connection to equipment operation and maintenance planning. Some metrics provide normalized values, generally in the range $[0,1]$, which mask the prediction time unit. Measures such as mean and median are commonly used to quantize the prediction error over a time period.
3. *Time-based penalization*. Decision-making is critical towards the end of life of a piece of equipment. Therefore, a penalization factor must be added to attribute greater importance to prediction errors made near the end of life. Usually, a weight function penalizes the error given the time instant at which the prediction is made. A linear function (5.2) and a Gaussian kernel (5.3) have been used as penalization functions (Saxena et al., 2008):

$$\alpha(t) = \frac{t}{\sum_{t=t_0}^{T} t}, \tag{5.2}$$

   where t_0 and T are the start and end of life, respectively, and

$$\alpha(t) = \exp\left(-\frac{(t-T)^2}{2(\frac{T}{2})^2}\right). \tag{5.3}$$

4. *Late prediction-based penalization*. Late predictions (positive Δ) are penalized more than early predictions (negative Δ) because of the impact on maintenance. Positive prediction errors are made by predicting a larger RUL value than the true RUL, causing the system to reach the end of life before maintenance is scheduled. Conversely, negative prediction errors favor the execution of early main-

tenance. Commonly, a function that receives a prediction error and retrieves a value related to its magnitude and sign is used. An exponential function (5.4) has been used as a penalization function:

$$\zeta(t) = \begin{cases} \exp\left(-\frac{\Delta}{\varphi_1}\right) - 1, \text{ if } \delta < 0 \\ \exp\left(\frac{\Delta}{\varphi_2}\right) - 1, \quad \text{if } \delta \geq 0 \end{cases}. \tag{5.4}$$

A summary of how these features are included in the most relevant accuracy metrics is presented in Table 5.2.

Table 5.2: Features of accuracy metrics

Metric	Overall Performance	Value in Time Units	Time Penalization	Late Prediction Penalization	References
Mean Absolute Error (MSE)	Yes	Yes	No	No	(Zhang et al., 2019; Zhu et al., 2018)
Exponential Transformed Accuracy (ETA)	No	Normalized	No	Yes	(Lei et al., 2018)
Relative Accuracy (RA)	No	Normalized	No	No	(Saxena et al., 2008)
Cumulative Relative Accuracy (CRA)	Yes	Normalized	Yes	No	(Saxena et al., 2008)
Mean Squared Error (MSE)	Yes	Yes	No	No	(Zeng et al., 2017; Di Maio and Zio, 2016)
Root Mean Squared Error (RMSE)	Yes	Yes	No	No	(Zeng et al., 2017; Di Maio and Zio, 2016; Lei et al., 2018)
Mean Absolute Percentage Error (MAPE)	Yes	Normalized	No	No	(Zeng et al., 2017; Di Maio and Zio, 2016; Zhang et al., 2019)
Sample Mean Error (SME)	Yes	Yes	No	No	(Zeng et al., 2017; Di Maio and Zio, 2016)
Sample Median Error (SMeE)	Yes	Yes	No	No	(Zeng et al., 2017; Di Maio and Zio, 2016)
Timeliness Weighted Error Bias (TWEB)	Yes	Normalized	Yes	Yes	(Zeng et al., 2017; Di Maio and Zio, 2016)

A key element for performance measurement and selection of RUL prediction methods is their accuracy as systems approach to the end of their useful life. Thus, time-based penalization is an important feature in accuracy metrics. Although penalization functions presented in (5.2) and (5.3) increase the penalization as time passes, they do not include information about the criticality of the maintenance task according to the specific application. Therefore, a new accuracy metric for evaluating the performance of RUL prediction methods, which includes the first three features presented in Table 5.2 and captures knowledge for supporting decision-making, is proposed in this chapter.

The first main contribution of this chapter is the proposal of a multivariate degradation model for predicting the RUL, which adapts to system conditions. In the construction of the model, only the variables that contribute significantly to the prognostic task are used. In addition, the reconfiguration of the prognostic model does not require a large amount of data. The second main contribution of this chapter is the proposal of a new performance metric for RUL prediction models that evaluates the prediction error throughout the degradation process in time units and considers task-specific knowledge for decision-making support.

The organization of the chapter is as follows. In Sect. 5.2, the new metric for the comparison of RUL prediction models is presented. Sect. 5.3 details the proposed method for predicting RUL. Sect. 5.4 validates the proposals by using the NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) benchmark (Saxena et al., 2008). Finally, the conclusions and several considerations for future research are presented.

5.2 Performance metric

Predictive maintenance tasks should be performed when the longest operating time of a system has been reached and before the end of its useful life to avoid unexpected stops of the production lines. Therefore, RUL prediction methods should be focused on minimizing the prediction error, fundamentally when a system approaches the end of its useful life. Thus, this chapter proposes a metric that periodically considers the error during the degradation process and includes time-based penalization, while keeping error values in time units.

5.2.1 Proposed performance metric

The proposed metric is defined below.

Definition 5.1. *Root Weighted Mean Squared Error*

Given a set of representative systems of the same type of system $\mathbf{E}$, $\{E_i \in \mathbf{E} : i = 1,2,...,N\}$ with cardinality N and each system consisting of a measurement set of size m with the respective time-stamped remaining useful life labels ($E_i = \{Y_i \in \Re^{m \times T}, RUL_i \in \Re^T\}$), the **Root Weighted Mean Squared Error** (*RWMSE*) for $\mathbf{E}$ is obtained as follows:

$$RWMSE = \frac{\sum_{i=1}^{N} RWMSE_i}{N}, \tag{5.5}$$

where

$$RWMSE_i = \sqrt{\frac{\sum_{t=1}^{T} \alpha_{RUL_{i,t}} \left(RUL^*_{i,t} - RUL_{i,t}\right)^2}{\sum_{t=1}^{T} \alpha_{RUL_{i,t}}}} \tag{5.6}$$

is a weighted average of the prediction errors during the degradation of system i, $RUL^*_{i,t}$ and $RUL_{i,t}$ are the predicted and true RULs at instant t during the degradation of system i, respectively; T is the total time over which the measurements were obtained; and $\alpha_{RUL_{i,t}}$ is the weight assigned to the RUL prediction error of system i at instant t.

The basis for this metric is to find a way to determine $\alpha_{RUL_{i,t}}$. This factor should penalize more the prediction error as the system approaches the end of its useful life and less when it is further away from

ceasing to work properly. Several types of functions could be used to determine $\alpha_{RUL_{i,t}}$. In this chapter, the exponential function given in (5.7) is selected, where a, b and c are the parameters of the function. This function places greater weight on prediction errors near the end of the systems life:

$$\alpha_{RUL_{i,t}} = a\exp(bx) + c. \tag{5.7}$$

The next step is to determine the parameters a, b and c of this function. In order to establish a relationship between $\alpha_{RUL_{i,t}}$ and $RUL_{i,t}$, x is defined according to the following equation:

$$x = RUL_{i,t} - RUL_w, \tag{5.8}$$

where RUL_w (w for warning) is the value of RUL from which the precision of the RUL prediction is considered critical, so the weight assigned to the error is greater from that moment on (see Fig. 5.1). In practice, this value can be defined by experts, or it can be assigned based on the knowledge acquired from data. The behavior of $\alpha_{RUL_{i,t}}$ corresponds to

$$\begin{cases} \alpha_{RUL_{i,t}} < a + c, \text{ if } RUL_{i,t} > RUL_w \\ \alpha_{RUL_{i,t}} = a + c, \text{ if } RUL_{i,t} = RUL_w \\ \alpha_{RUL_{i,t}} > a + c, \text{ if } RUL_{i,t} < RUL_w. \end{cases} \tag{5.9}$$

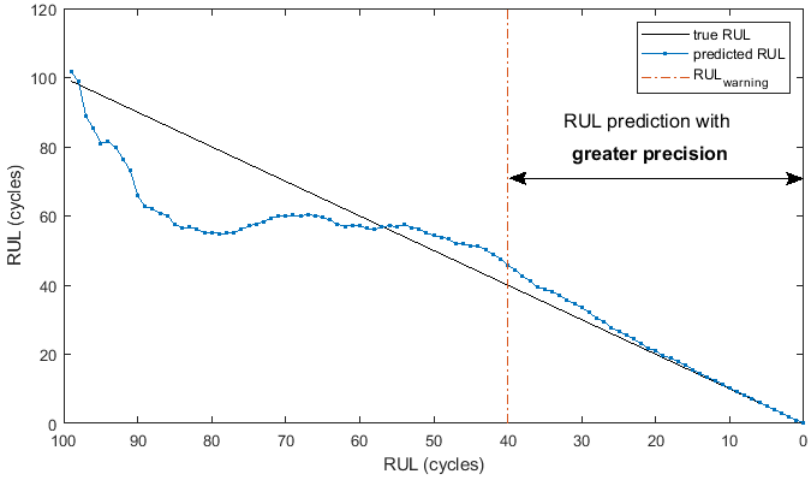


Fig. 5.1: Region of greater precision in the RUL prediction

Conditions on the parameters are $a \geq 0$ and $c = 0$ such that the effects of errors do not cancel each other and that each error contributes to the average, regardless of the time remaining from the prediction instant to the system end of life. Moreover, $a = 1$ to differentiate the weights assigned to both sides of RUL_w. By substituting the values assigned to the parameters in the inequalities (5.9), it is verified that the stated requirements are met:

$$\begin{cases} \alpha_{RUL_{i,t}} \geq 1, \text{ if } RUL_{i,t} \leq RUL_w \\ 0 < \alpha_{RUL_{i,t}} < 1, \text{ if } RUL_{i,t} > RUL_w. \end{cases} \tag{5.10}$$

The scaling parameter b remains to be set. In general, $\alpha_{RUL_{i,t}}$ should increase as the RUL decreases. Therefore, $b \in \mathbb{R}^{-}$. If $b = -1$ is considered, (5.7) would present a sudden change at RUL_w, assigning a very high weight when $RUL_{i,t} = 0$ and a very low weight to all errors when $RUL_{i,t} > RUL_w$. On the other hand, $\alpha_{RUL_{i,t}}$ should present a smooth shape, gradually varying as it approaches $RUL_{i,t} = 0$. Therefore, the smooth shape is guaranteed by selecting

$$b = \frac{-1}{RUL_w}, \tag{5.11}$$

where the condition $RUL_w > 1$ must be satisfied in order to obtain the desired behavior of the weight function.

Figure. 5.2 shows the behaviors of two weight functions with different values of b for three different RUL_w values.

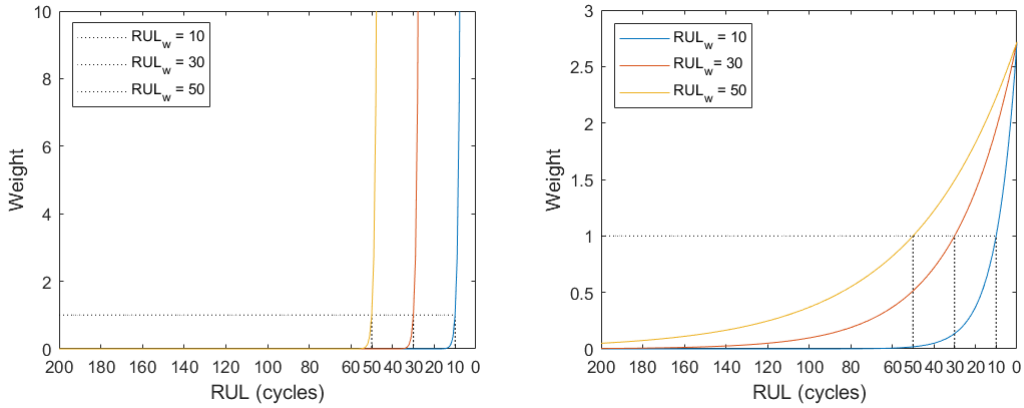


Fig. 5.2: Weight functions for $RUL_w = 10, 30, 50$: (a) $b = -1$ and (b) $b = \dfrac{-1}{RUL_w}$

Finally, the weight function obtained is

$$\alpha_{RUL_{i,t}} = \exp\left(1 - \frac{RUL_{i,t}}{RUL_w}\right). \tag{5.12}$$

Applying a weighted average approach to the prediction errors by using (5.12) as the weight function allows the $RWMSE$ metric to assess the performance of a model by taking into account the prediction accuracy during the entire degradation period. This metric allows the comparison of several prediction methods through a single value.

5.3 Procedure for RUL prediction

In this section, a procedure for RUL prediction is presented. Once a deviation in some characteristic parameter of the system has been detected, it is considered that the degradation of the system starts until

the end of its useful life. From the measurements of the system variables, it is possible to obtain a model for predicting how these variables will change with time based on the system behavior. A threshold for each variable in the system is defined to indicate the end of its useful life. The RUL of the system is determined by the time it takes for the variables reaching their corresponding predefined threshold. An advantage of the proposed degradation model is that it is adaptive. In this way, the accuracy of the RUL prediction may increase as more information is acquired. The proposed procedure is shown in Fig. 5.3. The configuration steps for this procedure are presented in Algorithm 1. A detailed explanation of each step will be given below.

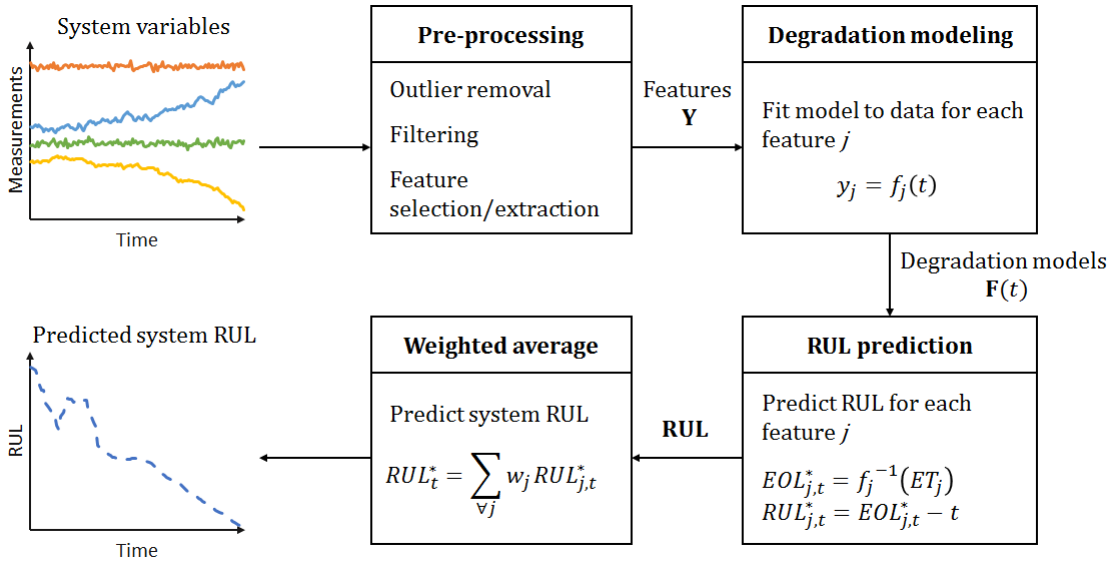


Fig. 5.3: Proposed procedure for RUL prediction

Algorithm 1 Off-line procedure

1. Apply *preprocessing* techniques to raw measurements of system variables to obtain representative features of the degradation process of the system.

for all features **do**

2. Select the *model structure* that best describes its behavior during the degradation process.

3. Define *constraints* for the selected model structure to improve degradation model prediction.

4. Obtain *EOL threshold* and *prediction model* for RUL.

end for

5. Obtain features weights for *weighted average* to predict the system RUL.

5.3.1 Preprocessing

The preprocessing stage is essential to obtain representative data of the degradation process. The data to be used in the configuration process should describe the degradation process from the moment a deviation from the nominal behavior is detected until the end of the system useful life. Preprocessing techniques are applied to extract the data relevant to the prediction model. Among the most used preprocessing techniques are:

1. Detection and removal of outliers
2. Filtering
3. Selection or extraction of features from the original variables.

In prognostics, modeling the degradation process of a system from data requires the features to reflect the progress of the degradation. If the behavior of a feature during the degradation process is stationary or constant, then it does not significantly help to predict the RUL. Therefore, features that are selected to model the degradation process should present progressive changes. Thus, metrics have been defined to select suitable prognostic features. Some of these metrics are *monotonicity*, *prognosability* and *trendability* (Coble, 2010). These metrics have been widely used to compare candidate prognostic features to determine which subset is most useful for individual-based prognostics (Coble and Hines, 2009; Coble, 2010; Ali et al., 2015; Liao et al., 2016; Saidi et al., 2017). Their values range from 0 to 1 where 0 indicates that the feature is not suitable for prognostics.

- ***Monotonicity*** quantifies monotonic trends as the system evolves toward failure (Coble, 2010). As a system gets progressively closer to failure, a suitable prognostic feature typically shows a monotonic trend. Conversely, any feature with a non-monotonic trend is less suitable. The formula to compute monotonicity is

$$Monotonicity = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{PD-ND}{T-l}\right|, \tag{5.13}$$

  where PD and ND are the number of positive and negative Δy_i differences, respectively; $\Delta y_i = y_i(t+l) - y_i(t)$ are the $T-l$ differences for each system; y_i represents the measurements of a feature of the system i; l is the number of data points used to calculate one difference; T is the number of instants of system i; and N is the number of systems.
- ***Prognosability*** is a measure of the variability of a feature at failure based on the trajectories of several run-to-failure experiments (Coble, 2010). A more prognosable feature has less variation at failure relative to the range between its initial and final values. As a system gets progressively closer to failure, a suitable prognostic feature is typically highly prognosable. Conversely, any feature that is non-prognosable is less suitable. The formula to compute prognosability is

$$Prognosability = \exp\left(-\frac{\sigma(y_i(T))}{\mu(|y_i(T)-y_i(1)|)}\right),\ \ i=1,...,N. \tag{5.14}$$

- ***Trendability*** is a measure of similarity between the trajectories of a feature measured in several run-to-failure experiments (Coble, 2010). A more trendable feature has trajectories with the same underlying shape. As a system moves progressively closer to failure, a suitable prognostic feature is typically

highly trendable. Conversely, any feature that is non-trendable is less suitable. The formula to compute trendability is

$$Trendability = \min_{i,j} \left| corr(y_i, y_j) \right|, \;\; i, j = 1, ..., N, \tag{5.15}$$

where

$$corr(y_i, y_j) = \frac{1}{T} \frac{\sum_{t=1}^{T} (y_{i_t} - \mu(y_i))(y_{j_t} - \mu(y_j))}{\sigma(y_i)\sigma(y_j)}. \tag{5.16}$$

When y_i and y_j have different lengths, the shorter vector is resampled to match the length of the longer vector. To facilitate this process, their time vectors are first normalized to percent lifetime: 0% to 100%.

By defining a fitness function as a weighted sum of the three metrics

$$fitness = w_m Monotonicity + w_p Prognosability + w_t Trendability, \tag{5.17}$$

the features can be selected. The constants w_m, w_p and w_t weigh the importance of each metric in the selection of the features.

5.3.2 Degradation modeling

For degradation modeling, gradual degradation under invariant operational conditions is assumed, as well as the system working under invariant operational conditions during its useful life. The system degradation model is based on the deterioration of each previously selected feature, as shown in Fig. 5.3. The parameters of the degradation model corresponding to each feature are adjusted according to the behavior of the variable by applying an on-line procedure (presented in Sect. 5.3.4) and also taking into account the model structure selected from the historical data set.

5.3.2.1 Model structure

A key element in modeling the behavior of each single feature is the selection of the model structure. Possible structures can range from the simplicity of a linear model to the complexity of a deep neural network. For single variable modeling, however, the selection should always focus on the simplest possible structure.

In this work, the following model structures are considered because of their simplicity:

$$\text{Linear: } f(t, \mathbf{p}) = y = p_1 t + p_2 \tag{5.18}$$

$$\text{Quadratic: } f(t, \mathbf{p}) = y = p_1 t^2 + p_2 t + p_3 \tag{5.19}$$

$$\text{Exponential: } f(t, \mathbf{p}) = y = p_1 e^{p_2 t} + p_3, \tag{5.20}$$

where y and t are the dependent and independent terms, respectively, and $\mathbf{p} = [p_1, p_2, p_3]$ is the vector of parameters of f to be fitted. As a measure of the error made in estimating a variable during the degradation process, the root of the mean squared error (RMSE) is used:

$$RMSE(y, \mathbf{p}) = \sqrt{\frac{\sum_{t=t_d}^{T} (f(t, \mathbf{p}) - y_t)^2}{T}}, \tag{5.21}$$

where t_d is the detection time of the degradation process, y_t is the expected value at instant t, and T is the total number of observations. Figure 5.4 shows an example of the response of the models obtained when fitting these structures to the data of an artificial feature.

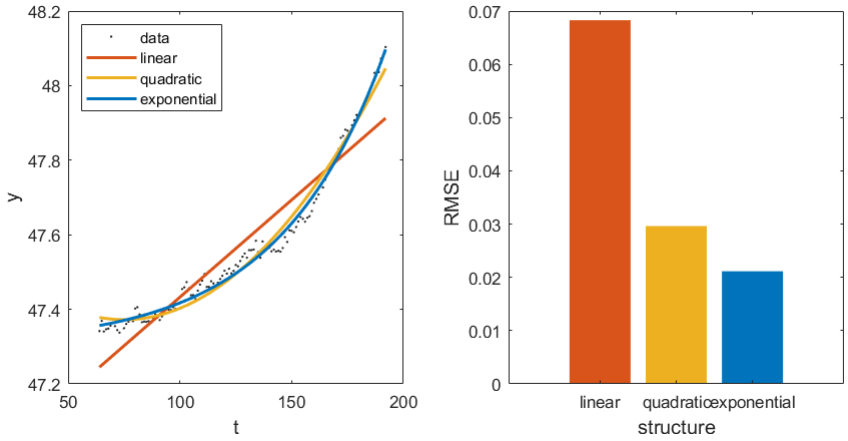


Fig. 5.4: Fit of different structures to the data of a variable

The structure for modeling each feature is selected independently since each feature may have a different evolution during the degradation of the system. Previous studies, which mainly focused on the construction of system health indices, present the exponential model on a recurring basis as part of the probabilistic models used to describe the health of systems that have cumulative degradation (Ali et al., 2018; Saidi et al., 2017; Dong et al., 2019).

5.3.2.2 Model constraints

Once a structure is selected, its ability to predict the future behavior of the degradation process is evaluated. While fitting a degradation model, both the fitting error of previous measurements and the prediction error of future measurements are considered in this chapter. If the prediction error increases, it would be necessary to constraint the parameter space of the model to reduce it. With the addition of regularization constraints to the fitting function, a search region is generated where two requirements are satisfied:

1. Minimize the model fit error for adapting to the current behavior of the feature during the degradation process and

2. Remove parameter values that introduce large errors in predicting future degradation behavior.

Fitting of the degradation model on-line is therefore presented as the following constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{p}} \ RMSE(y, \mathbf{p}) \\ &\text{s.t. } \mathbf{p} \in R_s, \end{aligned} \tag{5.22}$$

where y are the values of the features that represent the process degradation, R_s is the allowed region for parameter values of the selected structure. R_s is defined based on historical data of similar systems.

The simplest constraint defines constant lower and upper limits for each parameter p_i, creating what is called a box constraint. The type of constraint, however, may vary depending on the structure defined to model the degradation process. As the exponential structure is selected in this chapter to model the degradation process of each single feature, it is necessary to delve into the analysis of constraints for this type of structure.

Constraints for exponential structure

When analyzing the on-line response of the exponential degradation model, from the beginning of the degradation process to the end of the system's useful life, it is possible to see a wide range of model behaviors. Figure 5.5 shows how a model fits and predicts the behavior of an artificial variable at three instants during the degradation process ($t = [94, 124, 164]$).

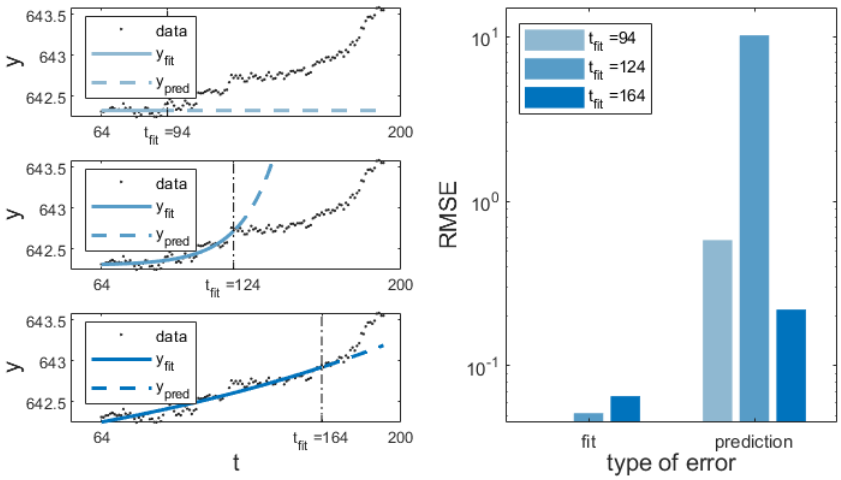


Fig. 5.5: On-line fitting of an exponential model to the data of one variable

The need for a constrained parameter space when the model of the feature is adjusted on-line will be explained based on Fig. 5.5. The fitting error for the feature shown validates the selection of the exponential structure. The error made in the prediction, however, is high, mainly at the beginning of the degradation process when there are few measurements of the variable. This is because the standard

performance measure (5.21) only focuses on minimizing the fitting error. At the start of the degradation process, the range of possible parameter values for which a good fit of the model is obtained is wide. A small fitting error for the initial data does not guarantee a small prediction error, as is the case of the model response for $t = 94$ and $t = 124$. The model response shows a constant behavior of the variable for $t = 94$, indicating that the system is not degrading. When the model is updated at time $t = 124$, the figure shows how the variable changes in response to the degradation process. The rate of degradation, however, is much higher than the true one. Both responses therefore describe behaviors of the variable that are not characteristic of the degradation process. When the acquired measurements of the variable reflect the entire progress of the degradation of the system, the range of possible values of the parameters decreases, and its response presents a better prediction. The primary objective of constraint enforcement is therefore to decrease the range of possible values for the degradation model parameters when a small amount of data is available.

The steps for configuring the constraints of the exponential structure (parameters p_1, p_2 and p_3 of Eq. (5.20)) are described below.

1. *Characterize the historical behavior of the feature.* It is necessary to adjust the parameters of the degradation model in such a way that its response adjusts as much as possible to the behavior of the feature through the entire degradation process.
   Since the exponential structure is nonlinear, it is proposed to use the trust-region reflective method (Coleman and Li, 1996b,a; Branch et al., 1999). Its main advantage is the speed of convergence. The convergence depends, however, on the search region of the parameters. Therefore, it is proposed to use the three-sigma procedure, presented in Algorithm 2, to reduce this region, starting from defining upper and lower limits for the parameters and to obtain the values of the parameters of the degradation model that characterize the historical behavior of the feature during the degradation process.
2. *Define the constraint.* By applying the procedure three-sigma setting $tolerance = 10^{-5}$, a first box constraint was obtained for the optimization problem stated in (5.22):

$$R_s = \{\mathbf{p} \in \Re^3 : p_{kL} \leq p_k \leq p_{kU}, k = 1,2,3\}. \tag{5.23}$$

   Once a constraint is defined, the fitting and prediction errors of the degradation model are evaluated.
   To define the type of constraint for each parameter, it is necessary to analyze the relationship between the parameters of the exponential structure (p_1, p_2 and p_3) and the behavior of the feature. Figure 5.6 shows the influence of each parameter on the exponential model. Parameters p_1 and p_2 determine the curvature of the model, so they are related to the rate of change of the feature. Similar systems can present different degrees of degradation, resulting in degradation models with different values of p_1 and p_2. The parameter p_3 represents the position of the asymptote on the y-axis. Values of the feature must be above or below this value, depending on the feature. Considering that the behavior of the feature is stationary during the normal operation of the system, it is possible to estimate the parameter p_3 from measurements acquired before the start of the degradation process. Estimating the value of the parameter p_3 from the start of the degradation process incorporates stability to the process of fitting the degradation model and, consequently, to the prediction of the RUL.
3. *Estimation of* p_3. The value of p_3 is estimated from the mean (μ) and the standard deviation (σ) of the measurements before the start of the degradation process. Since p_3 must function as an asymptote, three possible estimators were considered:

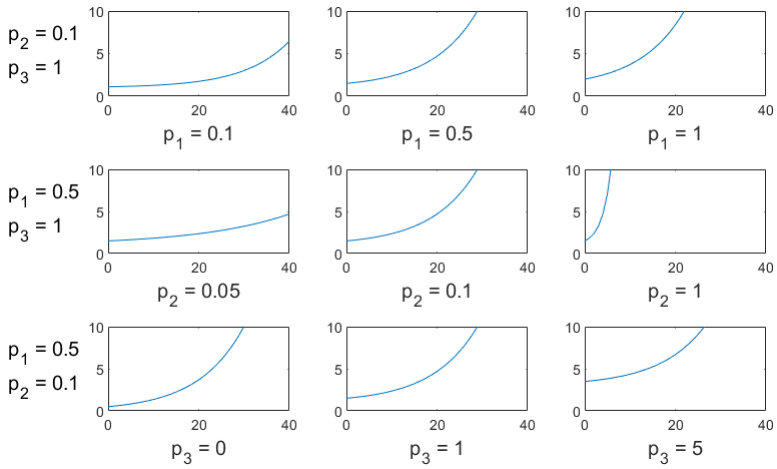


Fig. 5.6: Influence of the parameters p_1, p_2 and p_3 on the behavior of the model. Two parameters are held fixed in each row and the third parameter is varied

$$\mu + \sigma \text{ or } \mu - \sigma \tag{5.24}$$
$$\mu + 2\sigma \text{ or } \mu - 2\sigma \tag{5.25}$$
$$\mu + 3\sigma \text{ or } \mu - 3\sigma. \tag{5.26}$$

Algorithm 2 Procedure three-sigma to set the initial values of the degradation model parameters

1. Initialize the search region based on the relationship between parameters (p_k, $k = 1,2,3$) and feature measurements (y). To solve the optimization problem (5.22), the following constraint is considered:

$$\begin{aligned}(p_{1L_0}, p_{1U_0}) &= \begin{cases}(0,\text{inf}), & \text{if } y(T) - y(1) > 0\\ (-\text{inf},0), & \text{if } y(T) - y(1) < 0\end{cases}\\ (p_{2L_0}, p_{2U_0}) &= (0,\text{inf})\\ (p_{3L_0}, p_{3U_0}) &= (\min(y), \max(y))\\ R_s &= \{p_{kL_0} \le p_k \le p_{kU_0}\}.\end{aligned}$$

repeat

2. Obtain sets of parameters P_k by solving Eq.(5.22) considering R_s.

3. Create new limits:

$$\begin{aligned}p_{kL_{new}} &= \mu_{P_k} - 3\sigma_{P_k}\\ p_{kU_{new}} &= \mu_{P_k} + 3\sigma_{P_k}\\ R_s &= \{p_{kL_{new}} \le p_k \le p_{kU_{new}}\},\end{aligned}$$

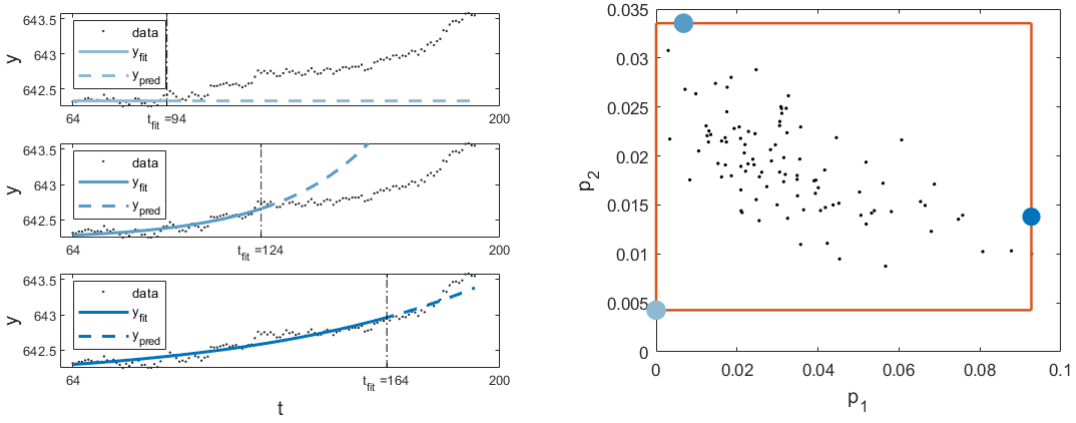
where μ_{P_k} and σ_{P_k} are the mean and standard deviation of the values of the set P_k.

until

$$\left|p_{kL_{new}} - p_{kL_{previous}}\right| < tolerance \quad \text{and} \quad \left|p_{kU_{new}} - p_{kU_{previous}}\right| < tolerance$$

according to the monotony of the feature during the degradation process. The estimation of the parameter p_3 from the beginning of the degradation process also allows reducing the complexity of the model.

4. *Constraint for p_1 and p_2.* From the values of the parameters of the degradation model for various systems, information is available about different degradation paths of similar systems, which is reflected in the values of the parameters p_1 and p_2. With the values of each parameter, a first approach is to define lower and upper limits for each parameter, creating what is called a box constraint. Figure5.7(a) shows the model response of an artificial variable obtained by fitting the parameters in three different instants of the degradation of a system. At the beginning of degradation process, when measurements contain little information, the model responses obtained show future degradation paths that differ significantly from those recorded.
   If the different values of the degradation model parameters obtained for various systems are plotted, in a two-dimensional space where each dimension corresponds to a parameter, as shown in Figure 5.7(b), the distribution of the parameters does not correspond to the box constraints defined above. The figure highlights three points that correspond to the values of p_1 and p_2 that generate the model responses shown in Figure 5.7(a). If the distribution of these points is analyzed, it is clear that the point corresponding to the model update at $t = 94$ is located in a corner of the box. The cluster of points shown in Figure 5.7(b) is considered the space that contains the values of the parameters p_1 and p_2 that characterize the degradation process of these systems. Any value of p_1 and p_2 generated by a point outside this cloud generates a model response that does not accurately represent the degradation process. Therefore, the constraint for p_1 and p_2 should be modeled using a region that conforms to the way the values are arranged in this space.



(a) (b)

Fig. 5.7: Fitting the exponential model with a box constraint

A model based on support vector data description (SVDD) is proposed to model the constraint for the parameters p_1 and p_2. SVDD allows creating a non-linear decision surface around a data set (Tax and Dui, 2004). Given a set of training data $\{\mathbf{x}_i\} \in \mathbb{R}^n$, $i = 1, ..., l$, the following optimization problem is solved:

$$\begin{aligned} &\min_{R,\vec{q},\vec{\xi}} \; R^2 + C\sum_{i=1}^{l}\xi_i \\ &\text{s.t. } \|\phi(\mathbf{x}_i)-\mathbf{q}\|^2 \leq (R^2+\xi_i),\;\; i=1,...,l \\ &\qquad \xi_i \geq 0,\;\; i=1,...,l, \end{aligned} \tag{5.27}$$

where $\phi(\mathbf{x})$ is a function mapping data to a higher dimensional space, $\xi = \{\xi_1,...,\xi_l\}$, ξ_i is the slack variable of i-th training sample, and $C > 0$ is a user-defined parameter. After (5.27) is solved, a hyperspherical model is characterized by a center $\mathbf{q}$ and a radius R. A testing instance $\mathbf{x}$ is detected as an outlier if

$$\|\phi(\mathbf{x})-\mathbf{q}\|^2 > R^2. \tag{5.28}$$

From the values of the parameters p_1 and p_2, a SVDD model characterizes its distribution and, therefore, the behavior of the feature during the degradation process. The SVDD model does not, however, guarantee that all points are within the hypersphere. Therefore, a threshold r_{th} is added to R so that all the degradation patterns are considered within the search region.
Taking into account the previous analysis, the constraint of the parameters p_1 and p_2 is defined by

$$R_s = \{\|\phi(\mathbf{p})-\mathbf{q}\| \leq R + r_{th}\}. \tag{5.29}$$

Figure 5.8(a) shows the degradation model responses obtained by fitting the parameters in three instants of the degradation of a system, using the constraint defined in (5.29). The system corresponds to that represented in Fig. 5.7(a). Figure 5.8(b) represents the SVDD model that defines the constraint, in which the points corresponding to the values of the parameters p_1 and p_2 are highlighted for the model responses shown in Fig. 5.8(a). The constraint allows bringing the model's response closer to the real behavior of the feature because as the model is updated by decreasing the fitting error to the acquired measurements, it is guaranteed that the values of the parameters p_1 and p_2 are within the region defined by the SVDD model.

5.3.3 RUL prediction

In this section, the method for predicting the RUL and the degradation model defined in the previous section is presented.

5.3.3.1 EOL threshold

The EOL threshold (ET), or failure threshold as it is also known, is a value of the feature at which point components or subsystems can be repaired or replaced immediately before the critical failure occurs (Bregon and Daigle, 2019). In most cases, the ET is estimated based on the physical knowledge of the system (Lei et al., 2016; Zhang et al., 2017; Yan et al., 2020). Expert knowledge, however, is not always accessible, and this value is often difficult to determine. To address this issue, some works consider

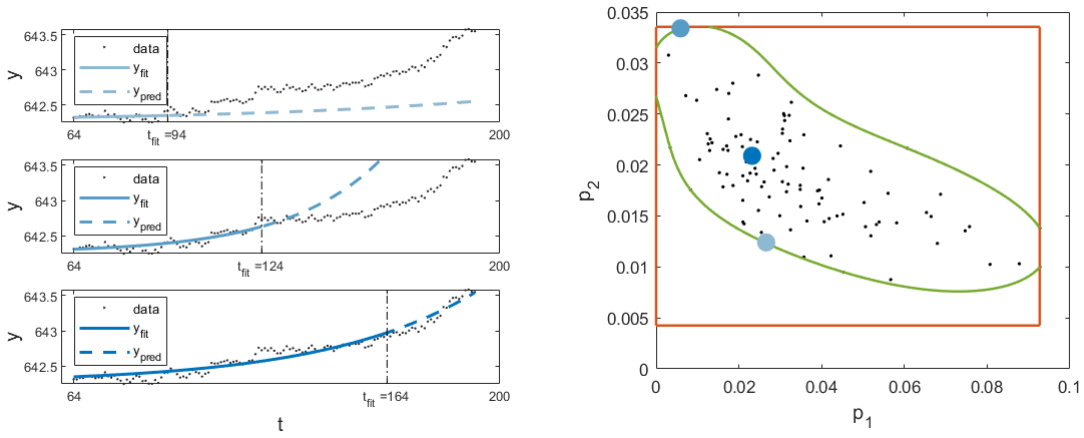


Fig. 5.8: Fitting the exponential model with the SVDD constraint

statistics like the mean of the last measurements of multiple systems as an estimation of the failure threshold (Liu et al., 2013). Since guaranteeing that these measurements follow a normal distribution for every type of system is difficult, the mean is not always an adequate estimator. In this chapter, the ET is estimated from the inverse cumulative distribution function of these measurements, with the assumption that they follow a unimodal distribution.

From the historical data of N systems, the last measurement of each variable y_T is obtained for each system. The ET is determined as the value with probability $1-p$ that the feature reaches the threshold during the system degradation:

$$ET = CDF^{-1}(p), 0 \leq p \leq 1, \tag{5.30}$$

where CDF^{-1} is the inverse cumulative distribution function of $y_{i,T}$, $i = 1,...,N$.

In the feature threshold configuration, several values of p must be considered according to the characteristics of the data. To determine which value of p is best to obtain ET, the RMSE is used as a measure of the prediction error of the EOL of the systems in the historical data:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (EOL^*_{i,T} - EOL_i)^2}{N}}, \tag{5.31}$$

where $EOL^*_{i,T}$ is the predicted EOL of the system i at the last moment of the useful life T, EOL_i is the true EOL of the system i, and N is the number of systems. The value of EOL_i is determined by the duration of the useful life of the system. The value of p selected to determine the threshold has the lowest prediction error.

In this work, the following definition for RUL included in the International Organization for Standardization (ISO) 13381-1:2015 is used; see Definition 5.2.

Definition 5.2. *Remaining useful life* . Remaining time before system's health falls below a defined failure threshold (ISO, 2015).

Therefore, once the EOL has been predicted, the RUL is determined according to (5.32) with t being the time at which the prediction is made. The periodicity at which the RUL is predicted must be determined, according to the criteria of experts.

Definition 5.3. *Remaining useful life of a system considering a single feature*. The remaining useful life of a system at continuous time t depending on a single feature y_j is

$$RUL^*_{j,t} = EOL^*_{j,t} - t \tag{5.32}$$

where

$$EOL^*_{j,t} = inf\{t_k \in \Re : t_k > t \ \wedge \ f_j^{-1}(t_k) = ET_j\}, \tag{5.33}$$

with ET_j being a performance threshold.

5.3.3.2 Prediction model

The prediction of the EOL at each moment during the process degradation can be performed by evaluating the degradation model in future instants until the estimated value exceeds the predefined threshold. This operation, however, is computationally expensive and may introduce errors in the prediction depending on the resolution of the degradation model evaluation. Conversely, if the inverse of the degradation model is obtained, the EOL can be predicted by evaluating the threshold.

By taking into account that the degradation process is modeled using an exponential structure and by evaluating (5.20) for the EOL prediction, $(t,y) = (EOL^*_{j,t}, ET_j)$, the following expression for ET_j is obtained:

$$ET_j = p_1 \exp(p_2 EOL^*_{j,t}) + p_3. \tag{5.34}$$

Thus, the prediction model is obtained by the inverse of the model represented in (5.34)

$$EOL^*_{j,t} = \frac{\ln(\frac{ET_j - p_3}{p_1})}{p_2}. \tag{5.35}$$

The parameters p_1, p_2 and p_3 correspond to those estimated for the degradation model. Figure 5.9 shows the response of the prediction model that allows checking the inverse behavior of the degradation model and visualizing the direct prediction of the EOL.

The explained procedure for the degradation modeling and the prediction of the EOL and the RUL is implemented for each feature.

5.3.3.3 Weighted average

The prediction model for each feature provides information on the progress of the degradation according to its effect on each one of them. The prediction of the RUL of the system should therefore reflect the contribution of each feature. The RUL of the system is then determined by the weighted average of the predicted RUL for each feature as established in (5.36).

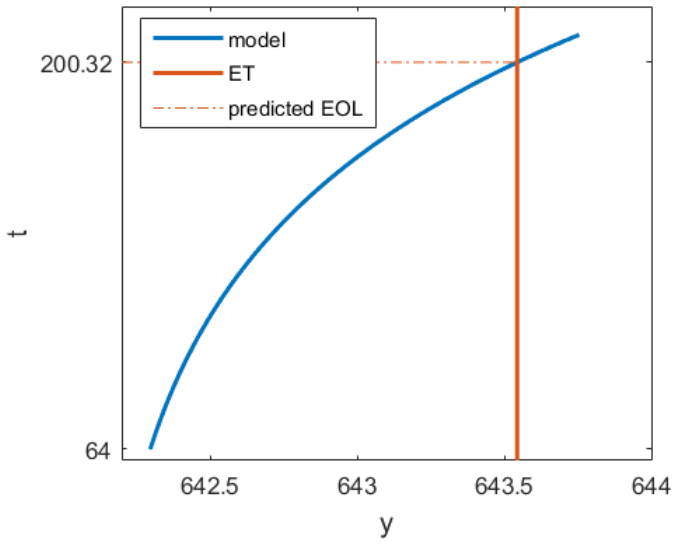


Fig. 5.9: Prediction model response

Definition 5.4. *Remaining useful life of a system considering a set of features*. The remaining useful life of a system i at continuous time t depending on a set of V features is

$$RUL^*_{i,t} = \frac{\sum_{j=1}^{V} w_j RUL^*_{j,t}}{V} \tag{5.36}$$

where $RUL^*_{j,i,t}$ is the RUL given feature j at time t and w_j is a weight scalar assigned to feature j according to its contribution in the prediction of the system's RUL.

Therefore, the next step in the configuration is to determine the contribution of each feature. The adjustment of the weights is presented as an optimization problem, where the objective is to minimize the RUL prediction error. The proposed performance metric, defined in (5.5), is used to calculate the RUL prediction error. The optimization problem is defined as

$$\begin{aligned} &\min_{w_j} \; RWMSE(RUL^*_{j,t}, RUL_{i,t}, w_j) \\ &\text{s.t.} \quad \sum_{j=1}^{V} w_j = 1 \\ &\qquad\quad w_j \leq 1, \forall j. \end{aligned} \tag{5.37}$$

In case of removing features from the weighted average, it is necessary to solve the optimization problem in (5.37) again in order to satisfy its constraints.

5.3.4 On-line procedure

After a configuration process, the prognostic procedure can be implemented to predict the RUL of a system. The on-line procedure for predicting the RUL of a system, presented in Algorithm 3, is implemented through a cumulative window approach. This approach requires the addition of two parameters: *minsize* or *minimum size* of the window of observations used to fit the degradation model, and *stride* or sampling time considered for updating the degradation model. As measurements are acquired, model parameters are recalculated to adapt to the current state of system degradation.

Algorithm 3 On-line procedure for RUL prediction

Parameters: *minsize* signifies *minimum size* of the window of observations required to estimate the parameters of a degradation model, and *stride* means sampling time considered for updating the degradation model

Inputs: t_d is the degradation detection time, R_s is the constrained region of the parameter space, and $\mathbf{w}$ is the weight vector that considers the importance of each feature for the accurate prediction of the system RUL

Initialization: $\mathbf{t}_{window} \leftarrow t_d$, $\mathbf{M}_{window} \leftarrow \emptyset$, $nobs \leftarrow 0$
repeat
  Acquire measurements $\mathbf{m}_t$ at time t
  $\mathbf{t}_{window} \leftarrow \mathbf{t}_{window} \bigcup t$
  $\mathbf{M}_{window} \leftarrow \mathbf{M}_{window} \bigcup \mathbf{m}_t$
  $nobs \leftarrow nobs + 1$
  if $nobs > minsize$ **then**
    Preprocess $\mathbf{M}_{window}$ to obtain features $\mathbf{Y}$
    for all $\mathbf{y}_j \in \mathbf{Y}$ **do**
      Obtain f_j given $\mathbf{M}_{window}$ by solving Eq. (5.22) considering R_s
      Predict EOL at current time $EOL^*_{j,i,t}$ for feature j
      $RUL^*_{j,t} \leftarrow EOL^*_{j,t} - t$
    end for
    Predict system RUL through Eq. (5.36) considering $\mathbf{w}$.
  end if
  $t \leftarrow t + stride$
until system failure

5.4 Application of the proposal

In this section, the application of the proposal to the NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) is presented (Saxena et al., 2008). Furthermore, the proposed model is compared with a state-of-the-art DNN-based model.

5.4.1 Case study

The C-MAPSS prognostics' case study is formed by four distinct data sets. Each one contains measurements of 21 sensors and three operating conditions. Each data set considers degradation information of several engines with both training and testing examples. To validate the proposal, the first data set containing information about one operating condition (altitude 0 ft, throttle angle 100 deg and Mach 0) and one type of failure (HPC degradation) is used. The data set contains a number of training engines (100 engines: training (N)) with run-to-failure information and a number of testing engines (100 engines: testing) with information terminating before a failure is observed.

Engines start operating with different degrees of initial wear but are considered healthy. As the number of cycles increases, each engine starts to deteriorate until failure. The training data set contains run-to-failure information for each engine. The testing data set contains temporal data up to a certain time before the engine fails. The objective is to predict the RUL of testing engines (Saxena et al., 2008).

In the experiments considered here, the training set is used for both training and testing/validating the procedure because the testing data set does not contain labeled information that can be used to verify the proposed procedure's performance.

5.4.2 Preprocessing

The training data set considered (FD001) is contaminated with noise, which makes analyzing the degradation trend difficult. The presence of noise in the measurements hinders the RUL prediction; thus filtering is applied as explained below.

Filtering. EWMA (exponential weighted moving average) filter is applied for removing noise. The expression that describes the filter is

$$\begin{aligned} y_0 &= x_0 \\ y_i &= \alpha x_i + (1-\alpha) y_{i-1} \end{aligned}. \tag{5.38}$$

where x_i represents the original data vector, y_i the filtered data vector, and α is the exponential percentage, defined as $\frac{2}{l+1}$, where l is the size of the window of observations to be filtered. For all variables, $l = 15$ is used.

Feature selection. Feature selection is performed by evaluating the fitness function in (5.17) for each variable and rejecting the variables with fitness below a predefined threshold. Monotonicity calculations for all variables use $l = 30$. The constants w_m, w_p and w_t for this application are set to 1 weighting equally for each metric. A threshold equal to 1.5 is selected since a sum is used as fitness function (Coble, 2010).

Figure 5.10 shows each metric and the fitness score for each variable. Sensors 1, 5, 6, 9, 10, 14, 16, 17, 18, and 19 have a fitness score under 1.5, which means that they are not very useful for prognostics. Therefore, these variables are not selected for modeling the degradation process.

Figure 5.11 shows the behavior of the variables used for modeling the degradation process for engine 1. These variables correspond to sensors 2, 3, 4, 7, 8, 11, 12, 13, 15, 20, and 21.

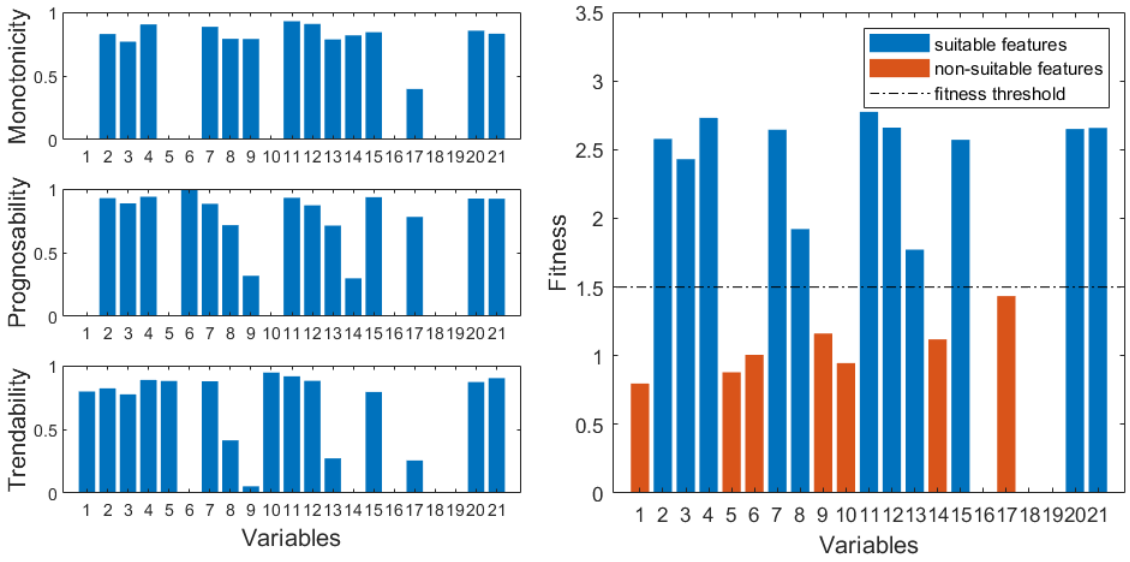


Fig. 5.10: Results of the feature selection process

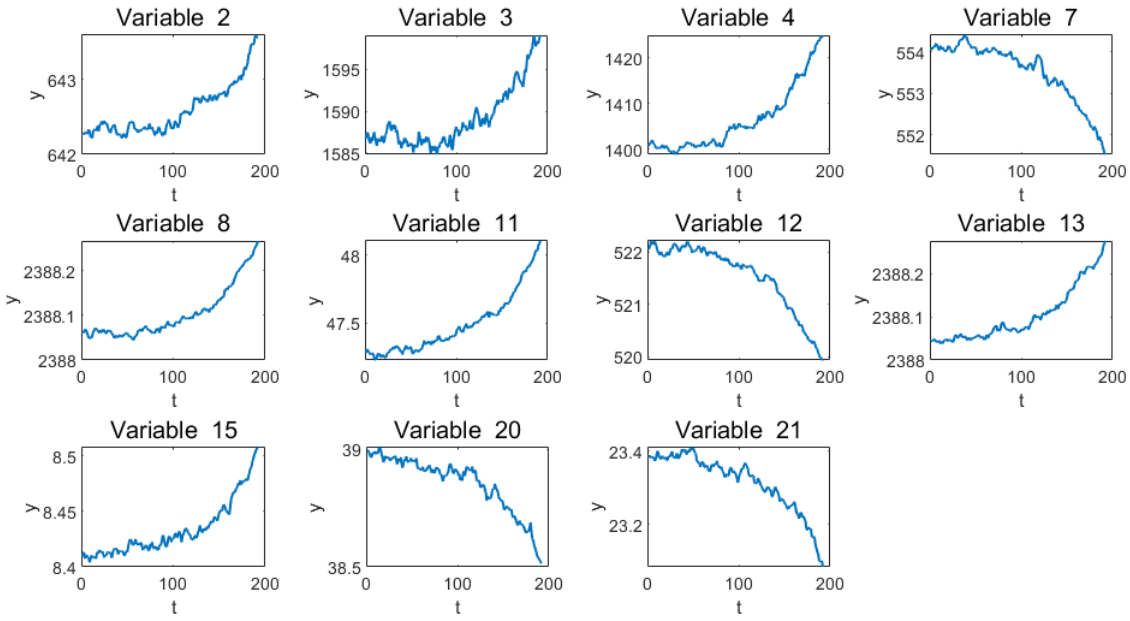


Fig. 5.11: Behavior of the selected variables to model the degradation process for engine 1.

5.4.3 Model training

Building each prediction model consists of selecting its structure as well as its constraints, establishing the EOL threshold, and estimating the weight of the variables for predicting the RUL of each engine.

5.4.3.1 Model structure

From the behavior of the variables selected in the preprocessing stage, the quadratic and exponential structures are considered for modeling the degradation process. Figure 5.12 shows the fitting error of both structures for the selected variables. The procedure presented in Sect. 5.3.2.2 is applied to fit the exponential structure with box constraints. The fitting error is averaged for each variable of the 100 engines of training data. The exponential structure is finally selected for modeling the degradation process because of its lower fitting error compared with the quadratic structure.

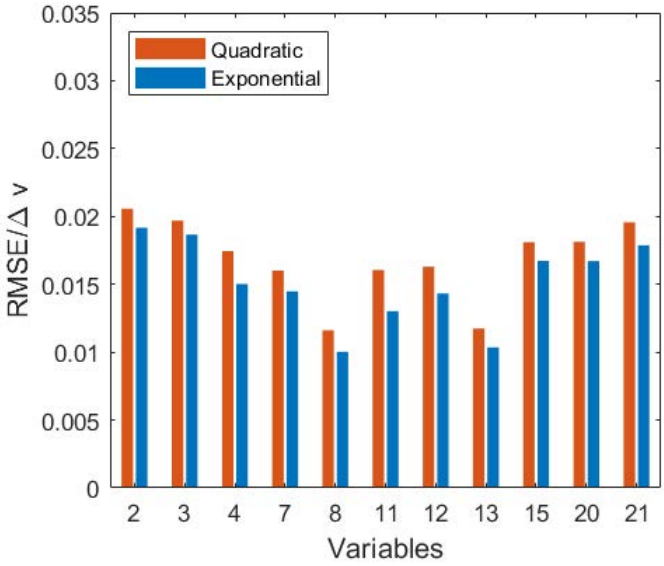


Fig. 5.12: Fitting error of the degradation models for each variable with quadratic and exponential structures. The error has been normalized with respect to the magnitude of each variable.

5.4.3.2 Model constraints

Estimation of p_3. For each estimator presented in (5.24), (5.25), and (5.26), the exponential structure prediction model of each variable was obtained. The fitting error of the degradation models (Fig. 5.13) shows that the best estimator for the parameter p_3 is $\mu + \sigma$ or $\mu - \sigma$, depending on the monotony of the variable.

Constraints for p_1 *and* p_2. The parameters obtained from the models in the estimation of p_3 are used to fit a SVDD model for each variable as a constraint for the parameters p_1 and p_2. In the SVDD model, the Gaussian kernel is used as the mapping function ϕ (5.39), with a parameter h that represents the bandwidth of the Gaussian function.

$$\phi(p_1, p_2) = \exp\left(-\frac{\|\langle p_1, p_2\rangle\|^2}{h^2}\right). \tag{5.39}$$

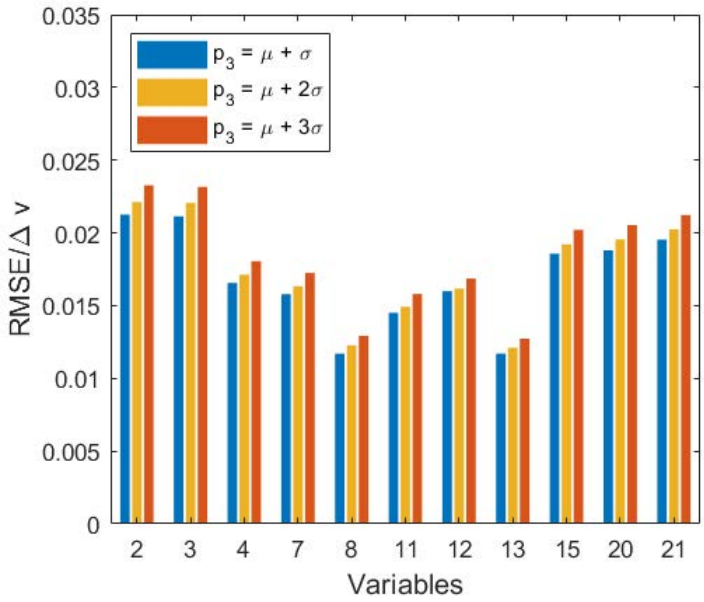


Fig. 5.13: Fitting error for the degradation models with different estimators of p_3. The error has been normalized with respect to the magnitude of each variable.

The distribution of the parameters for each variable and the SVDD model obtained are shown in Fig. 5.14. C and h in (5.27) and (5.39) for each SVDD model shown in Table 5.3. These parameters are manually tuned such that all models remain within the decision surface, excluding the space that does not contain previously fitted degradation models. Subsequently, the degradation models are readjusted to validate the defined constraints (Fig. 5.15). The fitting error obtained with the new constraints may vary with respect to the error obtained with the box constraints. It is important to emphasize that the acceptable fitting error depends on the user. In that case, tuning the constraints can be accepted in order to decrease the on-line prediction error.

Table 5.3: Values of the parameters of the SVDD model for each variable

Variable	2	3	4	7	8	11	12	13	15	20	21
C	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
h	2.3	2.7	2.4	2.3	2.5	2.3	2.3	2.3	2.6	2.3	2.2

5.4.3.3 EOL threshold

To select the threshold for each variable, the EOL prediction error is calculated for several values of p, by using all degradation measurements. The values of p are 0.1, 0.3, and 0.5 since these are values with which the probability of reaching the threshold is greater than or equal to 0.5. To this end, the EOL

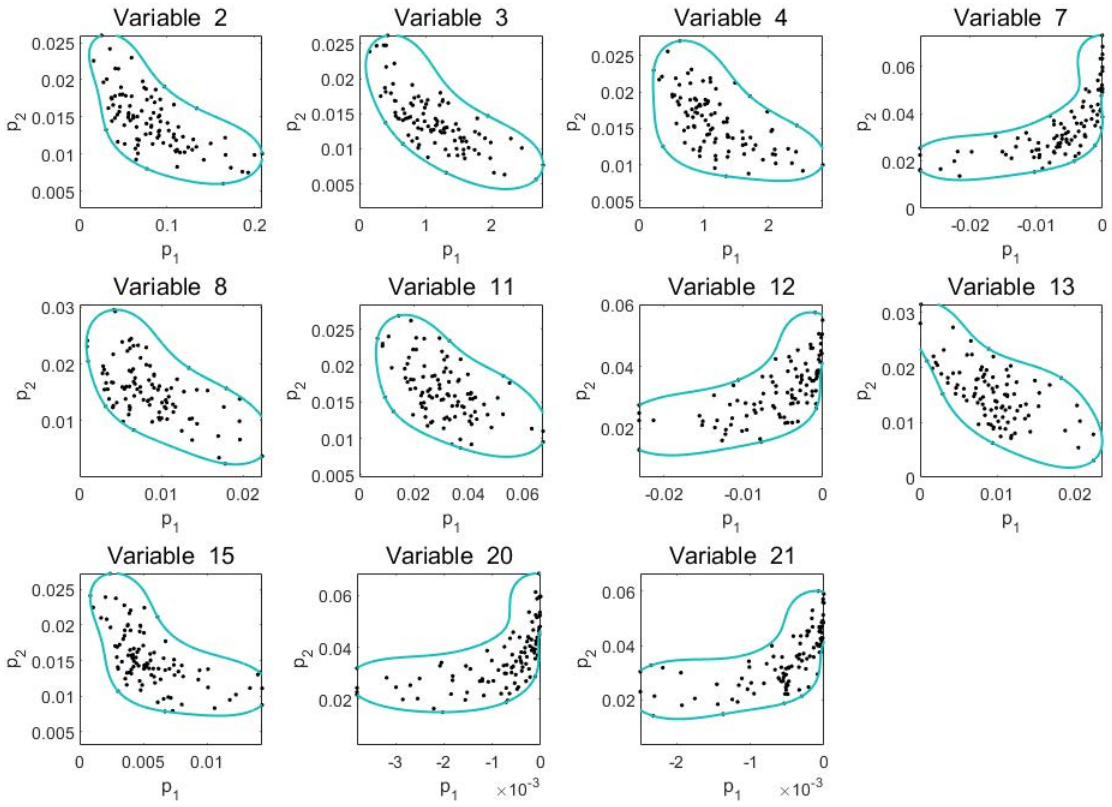


Fig. 5.14: Constraints of p_1 and p_2 for each variable.

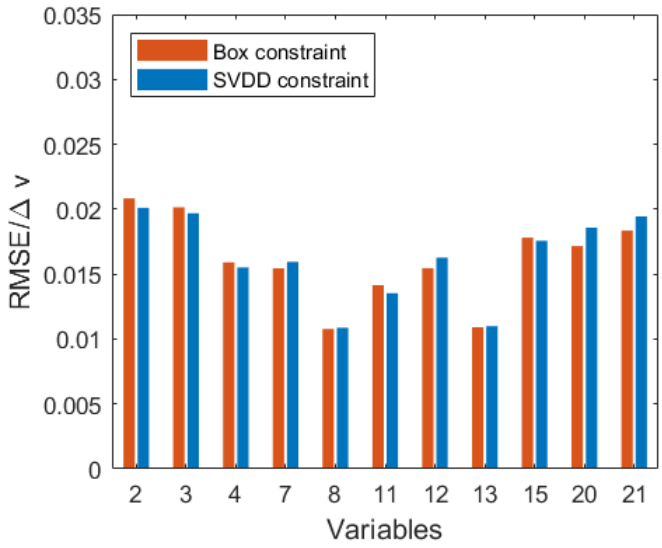


Fig. 5.15: Fitting error of the degradation models for each variable with box constraints and SVDD. The error has been normalized with respect to the magnitude of each variable

estimation error of the engines of the training set is averaged for each variable. From the results shown in Fig. 5.16, the criterion for determining the threshold for each variable is the value for $p = 0.5$.

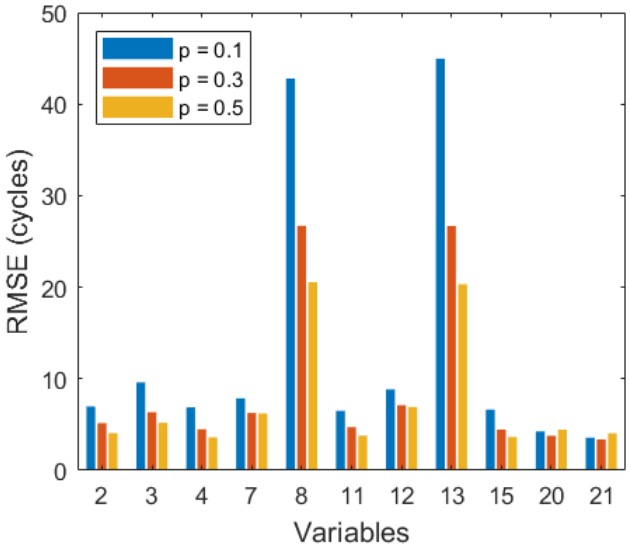


Fig. 5.16: EOL prediction error at the time of failure

5.4.3.4 Features weights

Adjusting the features weights requires evaluating the RUL prediction error on-line. To do this, the degradation models are fitted, and the RUL is predicted on-line for each variable of the 100 engines in the training set. A first step, which is recommended for the comparison, is to equitably assign weights to the variables and calculate the RUL prediction error. This value will serve as a reference for solving the optimization problem presented in (5.37), which allows obtaining the weights that minimize the RUL prediction error. Figure 5.17 represents the effect of the weight of the variables on the RUL prediction of the system by comparing the prediction error before and after the optimization of the weights. It can be observed that by optimizing the weights the combination of variables obtained significantly reduces the prediction error of the RUL.

From the weights presented in Fig. 5.17, the contribution of each variable in the prediction of the RUL can be determined. Thus, it is evident that variables 8 and 13 do not contribute significantly. These variables can be removed from the prognostics model, and, in this case, the weight of the remaining variables must be optimized again.

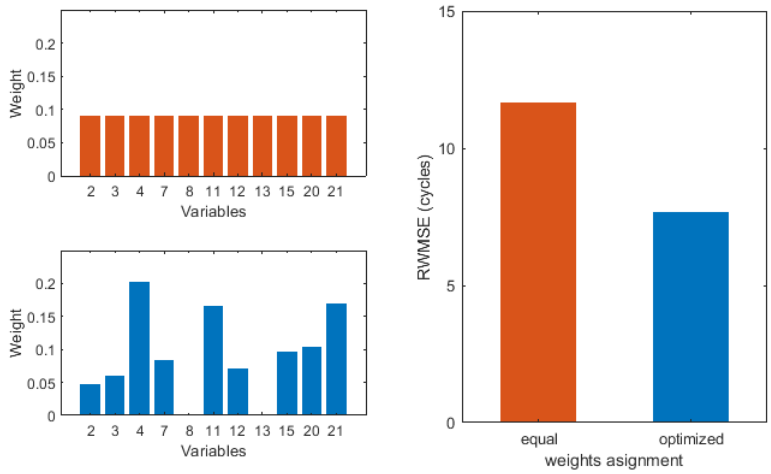


Fig. 5.17: Influence of the weight of the variables on the prediction of the RUL of the system

5.4.4 Model validation

During the construction of the prediction model, it is possible to evaluate each step using the training data. To validate the model, however, it is necessary to present data on the complete degradation process of engines not used in training. Therefore, the k-fold cross-validation method is used to validate the prediction model on the data of the 100 engines of the training set. For cross-validation, $k = 10$ is used, while also employing in each partition 90 training engines and 10 testing engines.

The proposed method for estimating RUL is compared with a DNN model (Song et al., 2018). This prediction model consists of the combination of two neural network structures: an autoencoder (AE) and a bidirectional long short-term memory neural network (BLSTM). By assuming the values presented in the chapter, the prediction model was implemented and the cross-validation method was applied. The same partitions were used for both prediction models.

For each partition, the off-line procedure presented in Algorithm 1 is used for the construction of the prediction model with the 90 engines in the training set. Algorithm 3 is implemented for the on-line RUL prediction of the 10 engines in the testing set. The *minimum size* for the cumulative window is set to 30, and the *stride* is set to 1, in order to compare the results with the DNN model.

The prediction error is calculated using the metric proposed in (5.5) with the value $RUL_w = [30, 40, 50]$. The prediction error for each partition is shown in Table 5.4.

The prediction errors of each model in each partition are compared by using the Wilcoxon statistical test. The significance level (α) of the test is 0.05. Table 5.5 shows the $p - value$ obtained when applying the statistical test. The test results, $p - value < \alpha$, show a significant difference between the prediction errors of both models, in which the proposed model in this chapter presents the smaller error.

The results shown in Fig. 5.18 demonstrate that the proposed model has better accuracy in the prediction of the RUL as the engines approach the end of their useful life.

Figure 5.19 shows the on-line fitting of the degradation model and the RUL prediction for variable 4 at three instants $t = [102, 132, 162]$ during the degradation process of engine 1. At each instant, the degradation model is updated with the acquired measurements, adapting to the engine conditions, and

Table 5.4: RWMSE. RUL prediction error of the proposed prediction model and the DNN-based model for each partition

	$RUL_w = 30$		$RUL_w = 40$		$RUL_w = 50$	
k	Proposal	BLSTM	Proposal	BLSTM	Proposal	BLSTM
1	8.81	10.51	10.11	12.32	10.92	12.97
2	8.56	10.30	10.54	12.44	12.01	13.32
3	7.36	8.59	8.87	10.56	9.98	11.43
4	8.81	12.66	10.30	14.75	11.30	15.92
5	7.45	10.51	8.97	12.60	10.10	13.66
6	8.92	9.56	10.93	11.68	12.41	12.91
7	10.52	10.25	12.99	12.36	14.83	13.67
8	7.14	10.66	7.98	12.08	8.56	12.83
9	10.27	12.27	11.58	15.16	12.75	16.52
10	6.67	14.21	7.80	15.85	8.59	16.52
average	**8.45**	10.95	**10.00**	12.98	**11.15**	13.98

Table 5.5: Results of applying the Wilcoxon test with $\alpha = 0.5$

	$RUL_w = 30$	$RUL_w = 40$	$RUL_w = 50$
$p - value$	1.16e-04	1.47e-04	6.92e-04

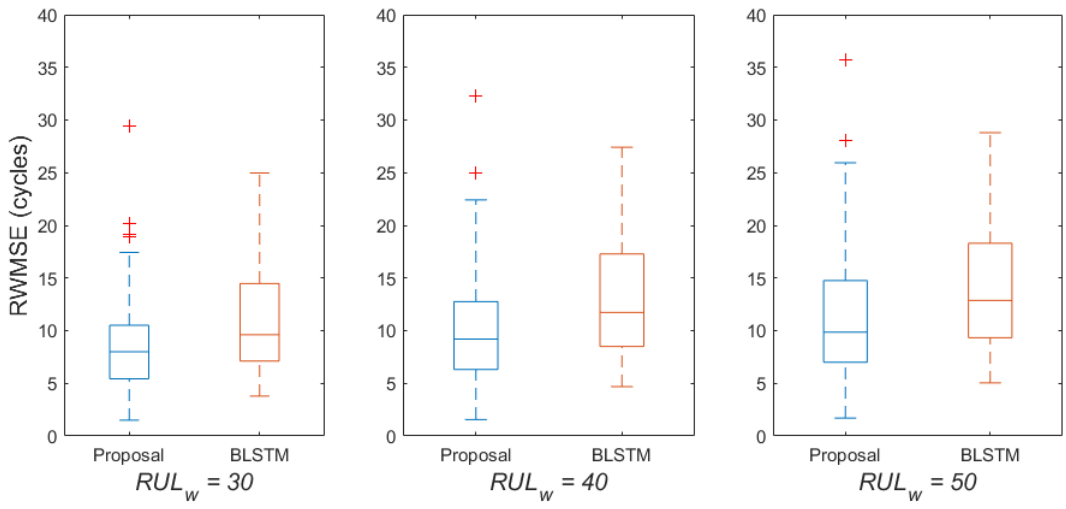


Fig. 5.18: Comparison of the RUL prediction error between the proposal and the DNN

progressively improving the RUL prediction. The predicted RUL for engine 1 throughout the degradation process, obtained from the weighted average of the predicted RUL for each variable, is presented in Fig. 5.20. The figure also presents the predicted RUL for engine 1 with the BLSTM neural network.

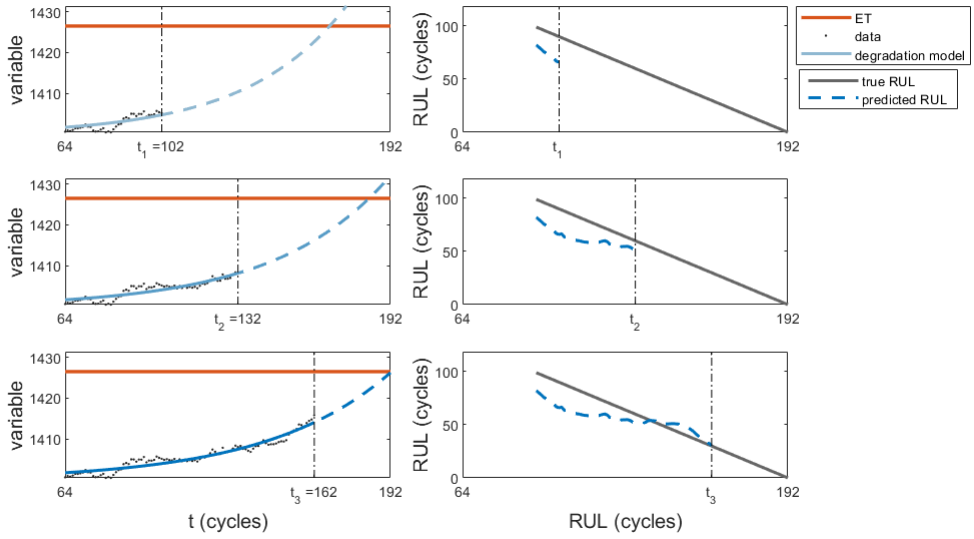


Fig. 5.19: On-line fitting and prediction for variable 4 of engine 1

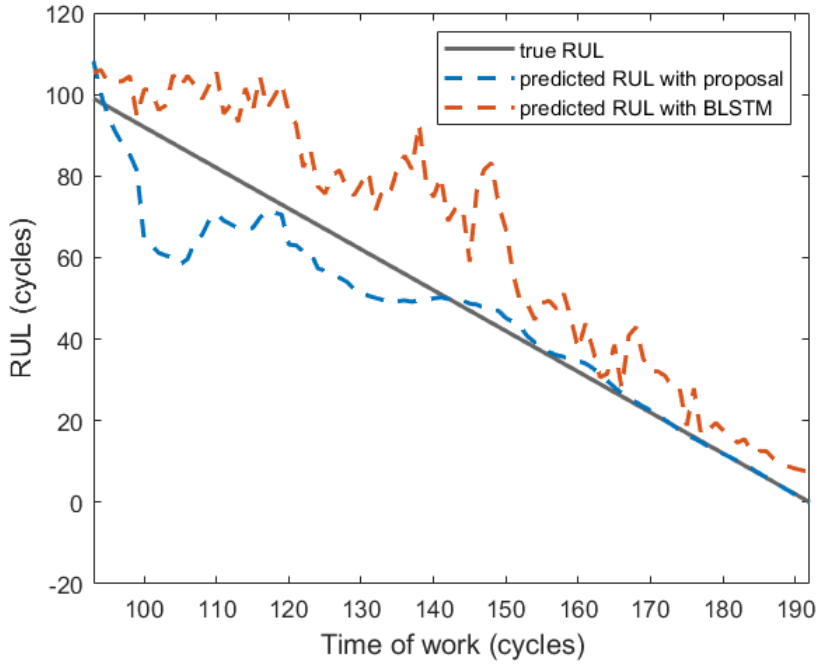


Fig. 5.20: RUL prediction for engine 1

5.5 Conclusion and future work

In this chapter, a new procedure is proposed to create RUL prediction models. To evaluate its effectiveness, the C-MAPSS data set is used, with exponential degradation models. The prediction model obtained is compared with a DNN-based model because of its widespread use in prediction tasks. Pre-

diction models are compared using a performance metric, RWMSE, which is proposed to evaluate RUL prediction models. This metric allows characterizing the evolution of the prediction error of the models and links the prediction of the RUL with the planning of maintenance tasks. The following conclusions can be stated based on the experimental results.

1. The prediction model adapts to the current conditions of the system during the degradation process, allowing to increase the prediction precision as the degradation progresses.
2. Constraints based on SVDD models allow accurate modeling of the historical behavior of the degradation process, which reduces the prediction error at the beginning of the degradation process when there are few measurements available.
3. Multivariable analysis determines the influence of each variable on the prediction of RUL and measures the importance of each variable for the RUL prediction process.

In future research, the procedure will be applied to other systems that reflects exponential behavior degradation. In addition, it will be extended to systems with other behaviors during their degradation process. Precision metrics for prognostics will be also analyzed.

Acknowledgements

Adriana Villalón-Falcón, Alberto Prieto-Moreno, and Orestes Llanes-Santiago acknowledge the financial support provided by Project No. 27, National Program of Research and Innovation ARIA from the Ministry of Science, Technology and Environment (CITMA), Cuba and Universidad Tecnológica de La Habana José Antonio Echeverría, CUJAE.

References

Ali, J. B., Chebel-Morello, B., Saidi, L., Malinowski, S., and Fnaiech, F. (2015). Accurate bearing remaining useful life prediction based on weibull distribution and artificial neural network. *Mechanical Systems and Signal Processing*, 56:150–172.

Ali, J. B., Saidi, L., Harrath, S., Bechhoefer, E., and Benbouzid, M. (2018). Online automatic diagnosis of wind turbine bearings progressive degradation under real experimental conditions based on unsupervised machine learning. *Applied Acoustics*, 132:167–181.

Branch, M. A., Coleman, T. F., and Li, Y. (1999). A subspace, interior and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing*, 21(1):1–23.

Bregon, A. and Daigle, M. J. (2019). Fundamentals of prognostics. In *Fault Diagnosis of Dynamic Systems*, pages 409–432. Springer.

Coble, J. and Hines, J. W. (2009). Identifying optimal prognostic parameters from data: a genetic algorithms approach. In *Annual conference of the prognostics and health management society*, volume 27.

Coble, J. B. (2010). *Merging data sources to predict remaining useful life–an automated method to identify prognostic parameters*. PhD thesis, University of Tennessee.

Coleman, T. F. and Li, Y. (1996a). An interior point trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445.

Coleman, T. F. and Li, Y. (1996b). A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058.

Di Maio, Franceso an Turati, P. and Zio, E. (2016). Prediction capability assessment of data-driven prognostic methods for railway applications. In *Proceedings of the third European conference of the prognostic and health management society*.

Dong, Q., Cui, L., and Si, S. (2019). Reliability and availability analysis of stochastic degradation systems based on bivariate wiener processes. *Applied Mathematical Modelling*, 79:414–433.

Ellefsen, A. L., Bjørlykhaug, E., Æsøy, V., Ushakov, S., and Zhang, H. (2018). Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. *Reliability Engineering & System Safety*, 183:240–251.

Feng, J., Kvam, P., and Tang, Y. (2016). Remaining useful lifetime prediction based on the damage-marker bivariate degradation model: A case study on lithium-ion batteries used in electric vehicles. *Engineering Failure Analysis*, 70:323–342.

Guo, L., Li, N., Jia, F., Lei, Y., and Lin, J. (2017). A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, 240:98–109.

ISO (2015). ISO 13381-1:2015 - Condition Monitoring and Diagnostics of Machines - Prognostics - Part 1: General Guidelines. Technical report, International Organization of Standardization.

Kim, N.-H., An, D., and Choi, J.-H. (2016). *Prognostics and health management of engineering systems: An introduction*. Springer.

Lei, Y., Li, N., Gontarz, S., Lin, J., Radkowski, S., and Dybala, J. (2016). A model-based method for remaining useful life prediction of machinery. *IEEE Transactions on Reliability*, 65(3):1314–1326.

Lei, Y., Li, N., Guo, L., Li, N., Yan, T., and Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical Systems and Signal Processing*, 104:799–834.

Li, N., Lei, Y., Lin, J., and Ding, S. X. (2015). An improved exponential model for predicting remaining useful life of rolling element bearings. *IEEE Transactions on Industrial Electronics*, 62(12):7762–7773.

Li, X., Ding, Q., and Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering and System Safety*, 172:1–11.

Liao, L., Jin, W., and Pavel, R. (2016). Enhanced restricted boltzmann machine with prognosability regularization for prognostics and health assessment. *IEEE Transactions on Industrial Electronics*, 63(11):7076–7083.

Liu, K., Gebraeel, N. Z., and Shi, J. (2013). A data-level fusion model for developing composite health indices for degradation modeling and prognostic analysis. *IEEE Transactions on Automation Science and Engineering*, 10(3):652–664.

Niu, G. (2017). *Data-driven technology for engineering systems health management*. Springer.

Saidi, L., Ali, J. B., Bechhoefer, E., and Benbouzid, M. (2017). Wind turbine high-speed shaft bearings health prognosis through a spectral kurtosis-derived indices and svr. *Applied Acoustics*, 120:1–8.

Saxena, A., Goebel, K., Simon, D., and Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pages 1–9. IEEE.

Song, Y., Shi, G., Chen, L., Huang, X., and Xia, T. (2018). Remaining useful life prediction of turbofan engine using hybrid model based on autoencoder and bidirectional long short-term memory. *Journal of Shanghai Jiatong University (Science)*, 23(1):85–94.

Tax, D. M. and Dui, R. P. (2004). Support vector data description. *Maching Learning*, 54(1):45–66.

Wu, Y., Yuan, M., Dong, S., Lin, L., and Liu, Y. (2018). Remaining useful life estimation of engineered systems using vanilla lstm neural networks. *Neurocomputing*, 275:167–179.

Yan, M., Wang, X., Wang, B., Chang, M., and Muhammad, I. (2020). Bearing remaining useful life prediction using support vector machine and hybrid degradation tracking model. *ISA transactions*, 98:471–482.

Yang, B., Liu, R., and Zio, E. (2019). Remaining useful life prediction based on a double-convolutional neural network architecture. *IEEE Transactions on Industrial Electronics*, 66(12):9521–9530.

Zeng, Z., Di Maio, F., Zio, E., and Kang, R. (2017). A hierarchical decision-making framework for the assessment of the prediction capability of prognostic methods. *Proceedings of the Institution of Mechanical Engineers, Part 0: Journal of Risk and Reliability*, 231(1):36–52.

Zhang, H., Chen, M., Xi, X., and Zhou, D. (2017). Remaining useful life prediction for degradation processes with long-range dependence. *IEEE Transactions on Reliability*, 66(4):1368–1379.

Zhang, H., Mo, Z., Wang, J., and Miao, Q. (2019). Nonlinear-drifted fractional brownian motion with multiple hidden state variables for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Reliability*, 69(2):768–780.

Zheng, S., Ristovski, K., Farahat, A., and Gupta, C. (2017). Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pages 88–95. IEEE.

Zhu, J., Chen, N., and Peng, W. (2018). Estimation of bearing remaining useful life based on multiscale convolutional neural network. *IEEE Transactions on Industrial Electronics*, 66(4):3208–3216.

This page intentionally left blank

Chapter 6

Leak localization in water distribution networks using machine learning based on cosine features

Ildeberto Santos-Ruiz[1], Francisco-Ronay López-Estrada[1], Vicenç Puig[2], and Guillermo Valencia-Palomo[3]

Abstract Locating leaks in water distribution networks is vital in drinking water management. This chapter presents a method for leak localization using pressure measurements modified by a nonlinear transformation related to direction cosines in the pressure space. Direction cosines are used as input variables in classifiers that locate leaks by associating them to the closest node in the distribution network. The feature extraction process is derived from hyper-parameter optimization with k-NN classifiers, but it is generalized to other classifiers based on supervised learning. The method was tested with physical measurements in some sectors of the Madrid network and synthetic data obtained by simulation with EPANET on a model of the Hanoi network. Some comparisons with other machine learning techniques, using raw pressures and pressure residuals, are performed to illustrate leak localization effectiveness when considering directional features.

6.1 Introduction

Loss caused by leakage is an underlying problem in drinking water management, as about one third of chemically treated water is lost globally because of leaks in distribution systems (OECD, 2016). Leaks must be detected and located promptly to be repaired in a short time and minimize the volume leaked. The leak localization problem becomes challenging because many leaks cannot be located visually because of the leaking liquid seeping underground instead of emerging to the surface.

The leak management process in a Water Distribution Network (WDN) consists of several stages. First, the leak must be perceived: the existence of water loss must be detected. If the leak is not visible, then a small region enclosing the leak (a leakage hotspot) must be estimated using calculations based on available hydraulic measurements. Second, to repair a leak, it must be accurately located using physical inspection and specialized devices, e.g., a geophone and correlators (Pilcher et al., 2007; Puig et al., 2017). The methods described in this chapter belong more to the prelocalization stage, which is intended

[1]Tecnológico Nacional de México / I.T. Tuxtla Gutiérrez, Carretera Panamericana S/N, 29050 Tuxtla Gutiérrez, Mexico,
e-mail: ildeberto.dr@tuxtla.tecnm.mx and frlopez@tuxtla.tecnm.mx

[2]Department of Automatic Control (ESAII), Universitat Politècnica de Catalunya UPC, Rambla de Sant Nebridi 10, 08222 Terrassa, Spain
e-mail: vicenc.puig@upc.edu

[3]Tecnológico Nacional de México, IT Hermosillo, Av. Tec. y Per. Poniente S/N, Hermosillo 83170, Mexico,
e-mail: gvalencia@hermosillo.tecnm.mx

to locate the closest nodes to the leak. Determining these nodes is helpful for the operators to locate the actual leak point in a short time. This task, however, is not trivial because of the following: small number of sensor measurements available compared with a large number of possible leak locations, the uncertainty in individual user consumption, and the noise and unpredictable transients that affect the certainty of the measurements. These difficulties, among others, make locating leaks a challenging problem; even with the results and achievements that will be discussed in the final sections, the leak localization problem remains open.

The proposed method is based on pressure residuals, which are the difference between nominal pressures (leak-free pressures, estimated from a network model) and current leaky pressures measured at specific network sensing nodes. A substantial difference from other leak localization methods is that the proposed method is not based on the magnitude of the residuals but on its direction, which is characterized by the so-called direction cosines.

The chapter first presents the state of the art that summarizes the most relevant previous works in this area. A simplifying hypothesis in these works is that single leaks occur at the nodes. Although leaks can appear anywhere in the network, this assumption simplifies the computation to estimate their position.

The leak localization problem is generally undetermined. Therefore, localization methods may fail to isolate the correct leaky node. It is possible, however, to identify candidate leaky nodes close to the actual leaky node, and it is essential to determine how good the predictability of each method is. Therefore, after presenting the literature review, some metrics will be described to evaluate a specific leak localization method and its effectiveness compared with other methods. The case studies used to validate the proposed methods will then be presented. Next, the methodology for leak localization considering both time-independent (steady-state) and extended period leak localization will be described. Finally, the results obtained in the case studies will be presented and discussed.

6.2 State of the art on leak localization in WDN

In the literature, there are different works reported in the context of leak detection and localization; for example, Pudar and Liggett (1992) formulated a solution based as an inverse problem of a steady-state hydraulic model by minimizing the differences between measurements and simulated states. By solving the inverse problem, the network model parameters, such as leak flow rate and location, are determined from the sensor measurements. However, the inverse problem is generally undetermined in an operational water network because of the small number of sensor measurements available.

To solve the indeterminate problem for leak localization when there are multiple equivalent leak candidates, Pudar and Liggett (1992) minimized the L_2-norm of the leak parameters. Conversely, Berglund et al. (2017) reduced the number of unknown measurements by selecting candidate nodes before solving the optimization problem. Recently, Blocher et al. (2020) addressed the inverse problem of leak localization using regularization methods to mitigate the ill-posedness effect. Using a regularization term, the authors solve the inverse problem to estimate leak hotspots by minimizing the sum of squared residuals, which represent the difference between measurements and steady-state model simulations. The performance of leak localization with this method depends on the regularization parameter choice and does not consider the uncertainty in pressures and residuals.

Sanz and Pérez (2015) and Sanz et al. (2016) also formulated an underdetermined inverse problem for demand calibration in order to solve the leak localization problem. To solve the ill-posed problem, Sanz and Pérez (2015) grouped the demand nodes based on the measured pressure sensitivity to a change in a node demand. The deviation of the calibrated demand from the expected demand for a given group indicates the presence and approximate location of a leak (Sanz et al., 2016). Prior assumptions about leak candidates and node groups are required, affecting the leak location search.

An alternative approach, which avoids the ill-posed inverse problem, directly compares the residuals with the pressure sensitivities for different leak locations (Casillas et al., 2013). Although this approach based on the sensitivities is limited to single leak scenarios, it has shown promising results in real applications (Pérez et al., 2011; Pérez et al., 2014).

The *pressure sensitivity matrix*, $\mathbf{S}$, expresses how sensitive the pressure at each node is to a leak occurring at any other node (even at the same node). Thus, the element s_{ij} that relates the leak flow rate q_i^ℓ at the i-th node with the pressure P_j at the j-th node is defined as $\frac{\partial P_j}{\partial q_i^\ell}$ so that

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \\ \vdots \\ \mathbf{s}_N \end{bmatrix} = \begin{bmatrix} \partial P_1/\partial q_1^\ell & \partial P_2/\partial q_1^\ell & \cdots & \partial P_N/\partial q_1^\ell \\ \partial P_1/\partial q_2^\ell & \partial P_2/\partial q_2^\ell & \cdots & \partial P_N/\partial q_2^\ell \\ \vdots & \vdots & \ddots & \vdots \\ \partial P_1/\partial q_N^\ell & \partial P_2/\partial q_N^\ell & \cdots & \partial P_N/\partial q_N^\ell \end{bmatrix}, \tag{6.1}$$

where N is the number of nodes in the network and $\mathbf{s}_i$ is the vector of sensitivities to the leak at the i-th node.[1] To locate the leak, each sensitivity vector $\mathbf{s}_i$ is compared with the residual vector at each time. The residual vector, $\mathbf{r}$, is defined by

$$\mathbf{r} = [P_1 - P_1^{\text{nom}}, P_2 - P_2^{\text{nom}}, \ldots, P_N - P_N^{\text{nom}}], \tag{6.2}$$

where P_j^{nom} indicates the nominal leak-free pressure at the j-th node, which is estimated by simulation using a hydraulic model of the network.

In practice, it is only necessary to compute the pressure sensitivity in the nodes with sensors because only the columns corresponding to the sensed nodes are retained. Similarly, only the residual components of the sensed nodes are calculated. On the other hand, given the difficulty of analytically calculating the partial derivatives in (6.1), the sensitivities are estimated numerically by increasing the leak flow rate from 0 to a given value Q^ℓ and measuring (by simulation) the corresponding increase in node pressures:

$$s_{ij} = \frac{\partial P_j}{\partial q_i^\ell} \approx \frac{\Delta P_j}{Q^\ell}. \tag{6.3}$$

One way to use the sensitivity matrix is to binarize it by setting a sensitivity threshold so that each sensitivity vector becomes a string of ones and zeros (Pérez et al., 2009, 2011). These binary strings constitute *leak signatures* and will be compared with the residual vector for matching. The residual vector is binarized in a similar way to the sensitivity matrix (6.1). If there is no exact match between the binary signature of a node and the binarized residual, the node with signature that has the highest number of bits matching the binarized residual is taken as the estimated position of the leak.

[1] Some authors use the transpose of the sensitivity matrix defined in (6.1) and the residual vector as a column vector.

Another proposal for locating leaks is through the statistical correlation of the actual residual with the vectors of the sensitivity matrix (Pérez et al., 2014):

$$\text{corr}(\mathbf{r}, \mathbf{s}_i) = \frac{(\mathbf{r} - \overline{\mathbf{r}})(\mathbf{s}_i - \overline{\mathbf{s}_i})^\mathsf{T}}{\sqrt{(\mathbf{r} - \overline{\mathbf{r}})(\mathbf{r} - \overline{\mathbf{r}})^\mathsf{T}}\sqrt{(\mathbf{s}_i - \overline{\mathbf{s}_i})(\mathbf{s}_i - \overline{\mathbf{s}_i})^\mathsf{T}}}, \tag{6.4}$$

where the overline $^-$ indicates the arithmetic means of the vector components. In this method, the sensitivity matrix is not binarized. Equation (6.4) aims to measure the similarity between the residual vector and the sensitivity vector of each leak node. The leak is assumed to be located at the node with sensitivity vector that has the highest correlation (closest to 1) with the residual vector. Framing it in machine learning methods, the proposal of Pérez et al. (2014) is about a 1-NN classifier with correlation distance.

A variant of the previous method was presented by Casillas et al. (2014), where the similarity is not measured with a correlation but through the angular closeness between the residual vector and the sensitivity vectors. The angle between $\mathbf{r}$ and $\mathbf{s}_i$ is given by

$$\alpha_i = \arccos\left(\frac{\mathbf{r} \cdot \mathbf{s}_i}{\|\mathbf{r}\| \, \|\mathbf{s}_i\|}\right), \tag{6.5}$$

where $\cdot$ denotes the inner product between vectors. The authors also proposed repeatedly calculating the angles along a time horizon (e.g., one day) to obtain an average angle. Finally, the leak estimated position is the node whose sensitivity vector has the smallest average angle. Framing it in machine learning methods, the proposal of Casillas et al. (2014) is about a 1-NN classifier with cosine distance.

An improvement in leak localization based on pressure residuals was presented by Casillas et al. (2015). They propose the residual vector projection towards a space of lower dimension (called *leak signature space*) in which the projections are less dependent on the leak magnitude and depend mainly on its location. The reduction in the residual space dimensionality is achieved by forcing one component of the residual vector to always be unitary. To locate the leak given a pressure residual, look for the smallest Euclidean distance between the actual projected residual and the leak signatures in the new space. Each leak signature is obtained by averaging the corresponding node's leak signatures. In terms of machine learning methods, the proposal of Casillas et al. (2015) is about an Euclidean 1-NN classifier with transformed residuals as features.

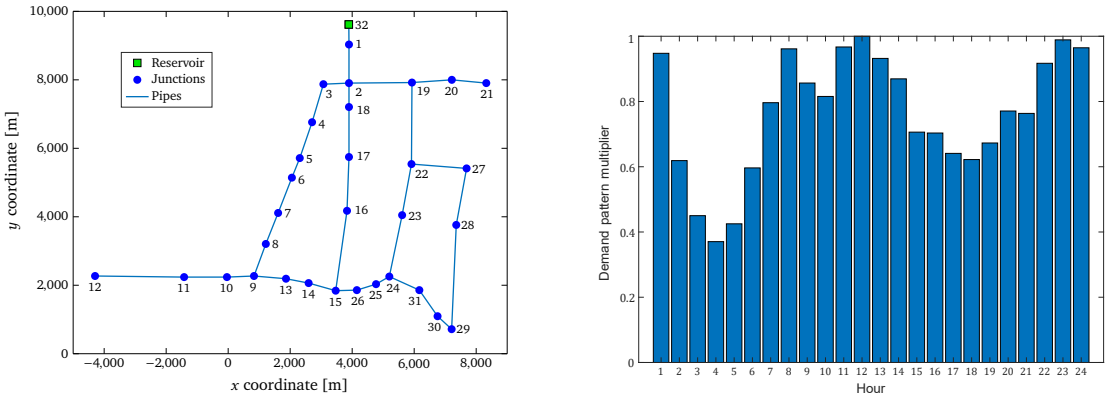
The correlation (6.4) and the angle between residuals (6.5) are the basis for discussing leak localization methods using classifiers in this chapter. Before addressing the problem of locating leaks using classifiers, the metrics used to evaluate the performance of different classifiers (or classifiers with different parameterization) will be presented.

6.3 Case studies

This section describes two water distribution networks used as case studies. The first case is a simple Hanoi network model, a network with few nodes widely used as a benchmark to evaluate various leak localization methods reported in the literature. Because of its small size, leak localization techniques can be comprehensively tested on this network. The second case is a Madrid network sector, a district

metered area (DMA), where leakage flow, node pressures, and inlet flow measurements were performed to test leak localization algorithms in real operating environments.

The Hanoi network model is composed of one reservoir, 31 junction nodes, and 34 pipes with a total length of 38.61 km organized in 3 loops, as shown in Fig. 6.1(a). No pumping facilities are considered since only a single fixed-head source (node number 32) at an elevation of 100 m is available. The Hanoi system was first presented by Fujiwara and Khang (1990) and is based on the planned trunk network of Hanoi, Vietnam. The data sets generated by simulation for this work consider leaks at all junction nodes with flow rates from 1 to 80 l/s, in addition to the leak-free nominal conditions.



(a) Geographical layout (b) Flow profile

Fig. 6.1: The Hanoi network

Because the Hanoi network model as published by Fujiwara and Khang (1990) does not include demand patterns of user consumption, a demand pattern of the Madrid DMA (described below) was used to modulate the base demand. At each time, the demand at the consumption nodes adjusts proportionally to the increase in the inflow, with variation throughout the day that is expressed by a multiplier coefficient. The flow profile used for the Hanoi network is shown in Fig. 6.1(b).

Synthetic data were obtained from the Hanoi network model using EPANET (Rossman et al., 2020; Eliades et al., 2016), which are node pressures of different leakage scenarios and at different times. The pressure data set contains a hypermatrix $\mathbf{P} \in \mathbb{R}^{24\times31\times1550}$ that captures the time variation of the node over the 24 hours of a day. One column is used for each node pressure, as shown in Fig. 6.2. The third dimension (depth) is used for the different leak scenarios. Each leak scenario refers to a combination of leak node and leak size. There are 1550 leak scenarios because 31 different leak nodes are combined with 50 different leak sizes (from 1 to 50 l/s). The data set also includes a vector $\mathbf{y} \in \mathbb{R}^{1550}$ containing the leak location for each scenario to be used in the classifier training. Additionally, the data set contains a pressured matrix $\mathbf{P}^{(0)} \in \mathbb{R}^{24\times31}$ with the information of the time variations of the node pressures under nominal leak-free conditions. The matrix $\mathbf{P}^{(0)}$ must be subtracted from each plane in $\mathbf{P}$ to obtain the pressure residuals used as features in leak classification.

The data obtained from simulations with the Hanoi network model are ideal since they consider a deterministic node demand without measurement noise. Therefore, Gaussian white noise is added to the simulated measurements under different levels. The added noise is specified by the signal-to-noise ratio

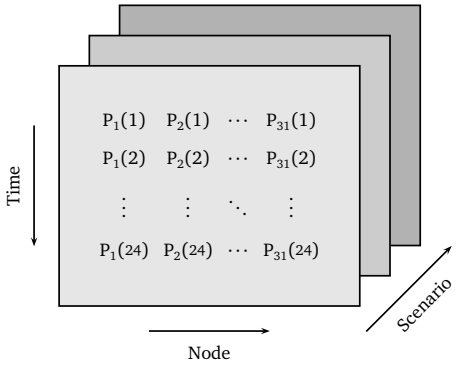
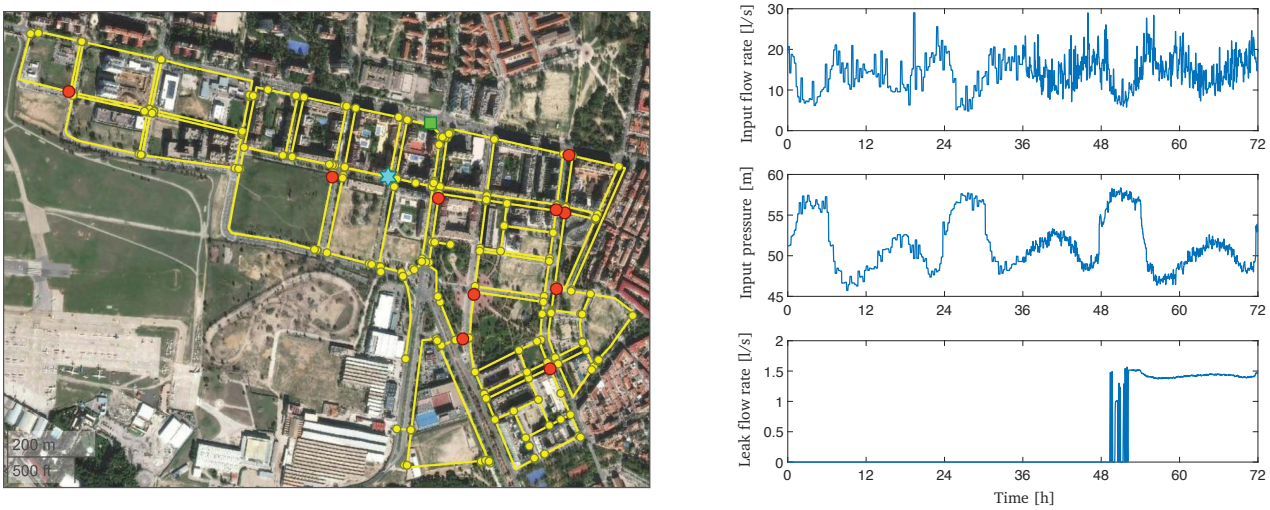


Fig. 6.2: Arrangement of the pressure hyper-matrix in the Hanoi data set

(SNR), which represents the size of the magnitude of the valid signal relative to the magnitude of the measurement noise, as defined by Box (1988):

$$\text{SNR dB} = 20 \log_{10}(\text{signal}/\text{noise}) . \tag{6.6}$$

The second case study is a DMA in the Madrid network formed by one reservoir and 312 junction nodes connected by 14224 m of pipe with diameters between 80 mm and 350 mm. The layout of this network is shown in Fig. 6.3, where the green square represents the reservoir, the 10 red circles represent installed pressure sensors, and the blue star indicates the leak node used in the tests. The mean pressure head is about 55 m, and the maximum pressure head available at the reservoir is about 58 m. Nevertheless, it varies throughout the day and is monitored by a sensor.



(a) Geographical distribution (b) Measurements for three days

Fig. 6.3: DMA in the Madrid network

Figure 6.3(b) shows the inlet flow and supply pressure of the network monitored during three days of operation, including a leak on the third day. The leak event is simulated by opening a fire hydrant at node

number 272. Figure 6.3(b) shows that generally, the supply pressure decreases when the consumption in the network increases.

The demand profile used to model user consumption on the Hanoi network was obtained from this DMA by averaging the network inflow over 10 consecutive days.

6.4 Performance metrics in leak localization

This section describes three indicators used in this work to quantify the performance of leak localization systems: confusion matrix, accuracy, and topological distance.

6.4.1 Confusion matrix

The confusion matrix is an instrument used in classification and pattern recognition to evaluate a classifier's ability to distinguish different classes or conditions of a system. This is why it is frequently used in FDI (fault detection and isolation) to assess a system/algorithm's ability to detect and isolate faults. In a classification problem with N classes, the confusion matrix is a $N \times N$ square integer matrix $\mathbf{C} = [c_{ij}]$, where c_{ij} contains the number of samples from the i-th class that are misclassified into the j-th class.

In leak localization, if each leak is associated with the closest node in the network, the rows of $\mathbf{C}$ represent the "true node" where the leak occurs (the node closest to the leak), while the columns of $\mathbf{C}$ represent the "predicted node" where leaks are located. Usually, it is represented by y the true leak node and by $\widehat{y}$ the predicted leak node (Fig. 6.4). Since the correct node (row) and the predicted node (column) should match in each test, ideally the confusion matrix should be diagonal. Nonzero items off the diagonal represent misclassifications.

Predicted leak node, $\widehat{y}$

True leak node, y	1	$\cdots$	j	$\cdots$	N
1	$c_{1,1}$		$c_{1,j}$		$c_{1,N}$
$\vdots$		$\ddots$			$\vdots$
i	$c_{i,1}$		$c_{i,j}$		$c_{i,N}$
$\vdots$				$\ddots$	$\vdots$
N	$c_{N,1}$		$c_{N,j}$		$c_{N,N}$

Fig. 6.4: Confusion matrix for leak localization

6.4.2 Classification accuracy

Although the confusion matrix provides a comprehensive summary of the classifier's ability to discriminate between leaks at different nodes, it is sometimes more convenient to summarize the overall performance in a single measure without detailing the separation between different leak classes. For this, the *classification accuracy* can be used, which is defined as the fraction of correctly classified samples in the classifier testing. In terms of the confusion matrix $\mathbf{C} = [c_{ij}]$, the accuracy (Acc) is obtained by dividing the sum of the elements on the main diagonal (the trace of $\mathbf{C}$) by the sum of all the elements:

$$\text{Acc} = \frac{\sum_{i=1}^{N} c_{ii}}{\sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij}}. \tag{6.7}$$

A value $\text{Acc} = 1$ indicates a perfect classifier performance, although in practice the accuracy is only close to 1 in the best cases. The proximity to 1, however, is only a good performance indicator when the test data set is balanced in the number of samples for each class. In leak localization tests, where classifiers are trained and tested on synthetic data, the simulations are programmed to equal the number of samples for each class (leak node).

In order to optimize leak localization, the performance of localization algorithms is often expressed in terms of *classification error* (classification loss), which is complementary to accuracy:

$$\text{Loss} = 1 - \text{Acc}. \tag{6.8}$$

A cross-validation strategy is also used to assess leak localization using classifiers in this work. The cross-validation procedure consists of partitioning the training data set into K groups and iterating K times, taking one of them as a test set and the rest as the training set. At each iteration, a classification loss is calculated; at the end, all losses are averaged to obtain a robust measure of performance called K-fold loss.

A limitation of the accuracy and the classification loss as performance metrics in leak localization is that they only consider the number of errors when locating leaks but not the magnitude of these errors. The following metric is intended to evaluate the performance of leak localization methods in a more comprehensive way.

6.4.3 Topological distance

The topological distance between two nodes is the number of links in the shortest path connecting them. Formally, a hydraulic network is a graph composed of a set of nodes $\mathscr{N} = \{N_k\}$ and a set of links $\mathscr{L} = \{L_k\}$, where each link (pipe) connects a pair of nodes. For any two nodes, i and j, the topological distance between them is denoted by d_{ij}. If the shortest path between this pair of nodes is represented by the link subset $\mathscr{P}_{ij} \subset \mathscr{L}$, then

$$d_{ij} = \left| \mathscr{P}_{ij} \right| \tag{6.9}$$

where $|\cdot|$ indicates cardinality.

In Definition (6.9), all links in the shortest path are equally valued; no link is considered more significant than another. It is also possible to define a length-weighted topological distance where the length of the pipe in each link of the shortest path is considered:

$$d_{ij} = \sum_{L_k \in \mathscr{P}_{ij}} \text{length}(L_k), \tag{6.10}$$

where $\text{length}(L_k)$ is the length (m) of the k-th link. Considering all the different pairs of nodes in a network, the topological distances are organized in a matrix. Since $d_{ij} = d_{ji}$, $\mathbf{D} = [d_{ij}]$ is a symmetric matrix. Furthermore, it has a null diagonal since $d_{ii} = 0$.

The topological distance between nodes is a metric for the leak localization error, using d_{ij} to quantify the error when a leak at node i is located at node j. It is also used as a cost function for hyperparameter optimization in training a classifier for leak localization. A series of data is frequently tested considering different leak scenarios, varying the leak location and magnitude. For each entry in the data set, a predicted leak node, $\widehat{y}$, is obtained and compared against the real leak node, y. A metric that summarizes the overall performance of the localization method on the test data is the *average topological distance* (ATD)

$$\text{ATD} = \frac{\sum_i \sum_j c_{ij}\, d_{ij}}{\sum_i \sum_j c_{ij}}, \tag{6.11}$$

where $[c_{ij}]$ is the confusion matrix described in Subsection 6.4.1.

Ideally, in the best case of leak localization, both ATD and classification loss should be 0, whereas the accuracy should tend to 1.

The topological distance between a pair of nodes can be obtained, among other methods, by Bread-First computation (Akiba et al., 2013) when all the links are treated with unit weight or by the Dijkstra algorithm (Dijkstra, 1959) when the pipe lengths are used to weigh the links.

6.5 Classification method based on Bayesian decision

Leak localization is presented in this work as a multiclass classification problem using supervised learning. The proposed classifiers can be framed within Bayes' decision theory and computed using probabilities. In this framework, the creation and tuning consist of proper probability density functions from training data. The Bayesian decision process is summarized below.

Let us consider a classification problem with J classes, determined by feature vectors $\mathbf{x} = [x_1, \ldots, x_n]$ representing observations of n features. The classifier response is a class label denoted by y, so y_j means that a given data sample belongs to the j-th class. To classify an observation $\mathbf{x}$, the *posterior probability* $P(y_j \,|\, \mathbf{x})$ must be estimated, which is the probability that the given observation belongs to the class y_j. It makes sense to assign an observation to the class with the highest posterior probability. Therefore, a way to estimate or fit these probabilities is required. Using the Bayes' theorem, the posterior probability can be computed as follows:

$$P(y_j \,|\, \mathbf{x}) = \frac{P(y_j)\, P(\mathbf{x} \,|\, y_j)}{P(\mathbf{x})}, \tag{6.12}$$

where

$$P(\mathbf{x}) = \sum_{j=1}^{J} P(y_j)\, P(\mathbf{x} \,|\, y_j) \tag{6.13}$$

is a common denominator for all classes and is called *evidence*. The factors $P(y_j)$ and $P(\mathbf{x} \,|\, y_j)$ in the above equations are the *prior probability* and *class-conditional probability* of class y_j, respectively. To distinguish between the concepts of prior and posterior probabilities, the rest of this section will use $P_0(\cdot)$ for prior and $\widehat{P}(\cdot)$ for posterior.

When there is no prior knowledge about the frequency of occurrence of each class, the prior probabilities $P_0(y_j)$ of each class can be assigned **uniformly** assuming that the cardinality of the classes is J and all members are equally probable:

$$P_0(y_j) = \frac{1}{\text{number of classes}} = \frac{1}{J}, \tag{6.14}$$

or it can be calculated **empirically** from the relative frequency of the class in the training set

$$P_0(y_j) = \frac{\text{number of samples in the } j\text{-th class}}{\text{total number of samples}}. \tag{6.15}$$

Once the prior probabilities and the class-conditional probabilities have been calculated, the predicted class is determined by the rule below.

Bayes decision rule

Given a feature vector $\mathbf{x}$, assign it to class y_j if

$$\widehat{P}(y_j \,|\, \mathbf{x}) > \widehat{P}(y_i \,|\, \mathbf{x}); \quad i = 1, 2, \ldots, J; i \neq j. \tag{6.16}$$

This means that the predicted class, denoted by $\widehat{y}(\mathbf{x})$, is given by

$$\widehat{y}(\mathbf{x}) = \arg\max_{j} \widehat{P}(y_j \,|\, \mathbf{x}), \tag{6.17}$$

which is also called the MAP (*maximum a posteriori*) decision rule.

In a diagnosis application, not all misclassifications have the same impact in the context of the problem. If some classification errors need to be weighted more than others, then the classification output can be computed by minimizing the *expected classification cost*,

$$\widehat{y}(\mathbf{x}) = \arg\min_{i=1,\ldots,J} \sum_{j=1}^{J} \widehat{P}(y_j \,|\, \mathbf{x})\, \Gamma_{ij}, \tag{6.18}$$

where Γ_{ij} is the cost of classifying an observation as class y_i when its true class is y_j. The simplest way to assign the cost of misclassification is

$$\Gamma_{ij} = \begin{cases} 0 & \text{if } i = j, \\ 1 & \text{if } i \neq j, \end{cases} \tag{6.19}$$

where all misclassifications are penalized equally; however, sometimes it is reasonable to use penalties other than 1. In applications for locating water leaks, where the classes are associated with the leak positions within the network, it is reasonable to increase the misclassification cost as the distance between the real leak position and the position predicted by the classifier.

All classification techniques used in this work can be framed within Bayes' decision theory, although they differ in estimating the class conditional probabilities. These techniques are naive Bayes classification, discriminant analysis, k-nearest neighbors (k-NN), and decision trees.

Naive Bayes classification and discriminant analysis compute the likelihood using Gaussian distributions. Classification by discriminant analysis assumes multivariate normal distributions for class-conditional probability densities, so the data are modeled using a Gaussian mixture distribution. Conversely, naive Bayes classification assumes that each feature x_i is independent of the other features, so the joint probability $P(\mathbf{x} \mid y_j)$ can be expressed from the product of the conditional probabilities of each feature individually.

In naive Bayesian classification and discriminant analysis, the distribution parameters (mean and variance/covariance) are estimated from the training data, so they are parametric classification methods. Furthermore, the classification by decision trees and k-NN is based on non-parametric models. The similarities/differences and the implementation details of each of these methods can be found in Martinez and Martinez (2015).

The Bayesian decision approach described in this section will be used both to estimate the leak location at a specific time point (Sections 6.7 and 6.8) and also to propagate the probability of leak occurrence between different time points (Section 6.9). The features x_i are obtained from the pressures in the nodes equipped with sensors, and the class labels y_j correspond to the leaky nodes.

Before describing the leak localization using classifiers, methods based on the sensitivity matrix are introduced since these have motivated the proposal.

6.6 Leak localization using the sensitivity matrix

Leaks in a network are manifested as pressure losses in the nodes. Therefore, the differences between the pressures before and after a leak event, the residuals, provide information for diagnosing the event. Leaks in the network, however, do not affect all nodes equally since the proportion in which the pressure residual of each node varies depends on the magnitude of the residual and its location.

From now on, matrices will be used to relate the leak position with the position of the pressure sensor where the residuals are computed. The leak node and the sensor node will be called i (row) and j (column), respectively. The matrix $[s_{ij}] = \partial P_j / \partial q_i$, as defined in (6.1), summarizes the pressure sensitivity at the j-th node to leakage at the i-th node. The heat map in Fig. 6.5 shows the sensitivity of all node pressures to leakage at each node in the Hanoi network using colors from white (lower sensitivity) to black (higher sensitivity), passing through yellow and red hues. It can be seen that nodes 21, 27, 28, 29, and 12 have high sensitivities, whereas node 1 has very low sensitivity. Furthermore, each node's pressure is more sensitive to leaks located at the same node. Therefore, the values on the matrix diagonal are dominant.

The sensitivity matrix, calculated by simulation, characterizes the influence of each possible leak on different node pressures. Thus, when a leak has already been detected, its location can be estimated by comparing the actual pressure residuals' similarity to the sensitivity vectors for each potential leaky node.

In order to measure the similarity between the residual vector and the sensitivity vectors, their Euclidean distance, their correlation, and the angle between them can be used. The Euclidean distance depends greatly on the magnitude of the leak and little on its location, so its use for leak localization requires that the residuals be mapped to a vector space where the magnitude dependence is minimized.

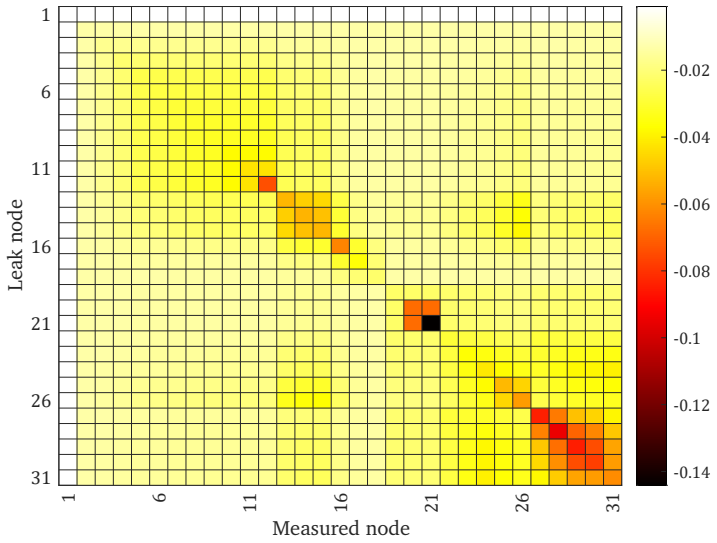


Fig. 6.5: Sensitivity matrix of Hanoi network

A similarity measure (between residuals and sensitivity vectors) little affected by the leak's magnitude is correlated. With this measure, the predicted leak node is the one with sensitivity vector $\mathbf{s}_i$ that has the highest correlation with the actual residual $\mathbf{r}$:

$$\widehat{y}(\mathbf{r}) = \underset{i\in 1,2,\ldots,n}{\arg\max}\ \mathrm{corr}(\mathbf{s}_i, \mathbf{r}). \tag{6.20}$$

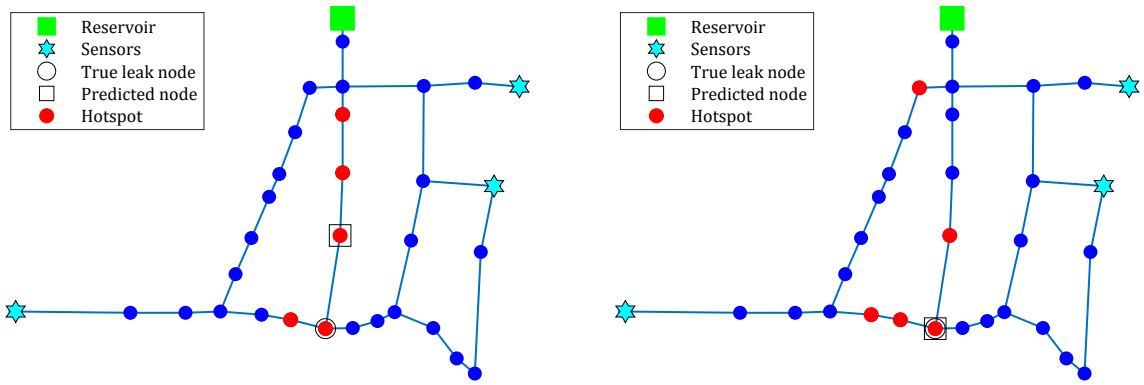
Since there is uncertainty in the leak node prediction, a list containing K nodes ($K > 1$) with the highest correlations in (6.20) can be considered to mark a "hotspot" of the leak in the network.

Another way to estimate the leak location is by computing the angle between the residual vector $\mathbf{r}$ and each of the sensitivity vectors $\mathbf{s}_i$. The node i with the minimum angle between $\mathbf{s}_i$ and $\mathbf{r}$ is assumed as a leak node. In practice, it is not necessary to compute the angle α_i between $\mathbf{s}_i$ and $\mathbf{r}$ but only the cosine of this angle. The node for which the cosine is maximum is assumed as the leak node (an angle is a minimum if its cosine is maximum):

$$\widehat{y}(\mathbf{r}) = \underset{i\in 1,2,\ldots,n}{\arg\min}\ \alpha_i = \underset{i\in 1,2,\ldots,n}{\arg\max}\ \cos(\alpha_i) = \underset{i\in 1,2,\ldots,n}{\arg\max}\ \frac{\mathbf{r}\cdot\mathbf{s}_i}{\|\mathbf{r}\|\,\|\mathbf{s}_i\|}. \tag{6.21}$$

A list containing K nodes ($K > 1$) with the highest cosines in (6.21) are considered to mark a hotspot for the tested leak.

Figure 6.6 shows the results of locating a leak in the Hanoi network using both methods: maximum correlation and minimum angle. As can be seen, the maximum correlation method does not locate the exact node of the leak but estimates a consistent hotspot. In contrast, the minimum angle method locates the exact leak node but estimates a more dispersed hotspot. A sensitivity matrix with $\Delta q = 25$ l/s was used. For the localization test, a leakage of 18 l/s was simulated at node 15 and noise of SNR = 60 dB added to the pressure sensors at nodes 12, 21, and 27.



(a) Prediction by maximum correlation (b) Prediction by minimum angle

Fig. 6.6: Leak locations obtained by maximum correlation and minimum angle in the Hanoi network

The maximum correlation and minimum angle methods have a common characteristic: they find the training sample with the most significant similarity (the minimum correlation/cosine distance). A leak localization method using k-NN classifiers is considered in the next section to generalize this approach to other distance metrics.

6.7 Leak localization using k-NN with hyper-parameter optimization

The k-NN (k-Nearest Neighbors) method classifies test data by taking the most frequent class among the k feature vectors most similar to the test data within the training set. In the simplest form of k-NN, where $k = 1$, only the sample from the training set that is closest to the test sample (nearest neighbor in feature space) is used. The test sample is then assumed to belong to the same class as the nearest training sample. Different distance metrics can measure the similarity between two feature vectors, including the well-known Euclidean distance.

The measure used to quantify the nearness between points significantly influences the k-NN classifier's performance. Although Euclidean distance is the most common way to measure dissimilarity between data, it is also possible to use other dissimilarity measures that express other types of distance between points in feature space. Some frequently used distance metrics are listed in Table 6.1, where $\mathbf{x}$ and $\mathbf{x}'$ are assumed to be feature vectors in $\mathbb{R}^n$.

Table 6.1: Distance metrics in k-NN classification

Distance name	**Definition**
Euclidean distance	$d(\mathbf{x},\mathbf{x}') = \sqrt{(\mathbf{x}-\mathbf{x}')^{\mathsf{T}}(\mathbf{x}-\mathbf{x}')}$
Mahalanobis distance	$d(\mathbf{x},\mathbf{x}') = \sqrt{(\mathbf{x}-\mathbf{x}')^{\mathsf{T}}\mathbf{S}^{-1}(\mathbf{x}-\mathbf{x}')}$
Manhattan (city block) distance	$d(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{n}\lvert x_i - x_i'\rvert$
Chebyshev distance	$d(\mathbf{x},\mathbf{x}') = \max_i \lvert x_i - x_i'\rvert$
Minkowski distance	$d(\mathbf{x},\mathbf{x}') = (\sum_{i=1}^{n}\lvert x_i - x_i'\rvert^p)^{1/p}$
Correlation distance	$d(\mathbf{x},\mathbf{x}') = 1 - \mathrm{corr}(\mathbf{x},\mathbf{x}')$
Cosine distance	$d(\mathbf{x},\mathbf{x}') = 1 - \cos(\mathbf{x},\mathbf{x}')$

The Manhattan, Euclidean, and Chebyshev distances in Table 6.1 are special cases of the Minkowski distance for values $p = 1$, $p = 2$, and $p = \infty$, respectively. The matrix $\mathbf{S}$ on the Mahalanobis distance is the data covariance (De Maesschalck et al., 2000). On the other hand, $\mathrm{corr}(\mathbf{x},\mathbf{x}')$ is the Pearson's linear correlation between $\mathbf{x}$ and $\mathbf{x}'$, treated as sequences of values, and $\cos(\mathbf{x},\mathbf{x}')$ denotes the cosine of the angle between the $\mathbf{x}$ and $\mathbf{x}'$ vectors in the n-dimensional feature space, as defined in (6.5). A cosine value of 0 means that the two vectors are orthogonal and have no match. The closer the cosine value is to 1, the smaller the angle and the greater the match between vectors.

Fine-tuning a k-NN classifier should include determining the most appropriate distance metric for the problem and determining the number of neighbors (k) to consider. The distance metric and the number of neighbors are known as k-NN *hyperparameters*.

The performance of a k-NN classifier depends on the number of nearest neighbors to use. Small values of k tend to produce an overfitted classifier that is very sensitive to measurement noise in features. In general, k-NN considers a higher number of neighbors, $k > 1$, which allows a more robust classification (less sensitive to outliers and measurement noise).

The general idea of leak localization using classifiers is to fit a predictive model that estimates the leak node location (class label) based on a set of node pressures (features) measurements. Next, leak location is explored in this section using k-NN as the first classification method. Unlike the methods described in the previous section, the k-NN classification generalizes the leaky node search by considering any number of nodes with similar leaks and for any distance metric.

From the Hanoi network pressure data set, where leaks of different flow rates are considered in each node of the network, k-NN classifiers were trained considering two different sets of features. In the first set, the pressures measured at the nodes were used directly, whereas in the second set the pressure residuals were used. For each set of features, the effect of varying the number of neighbors has been analyzed as well as the distance metric used to measure the nearness between neighbors. The results presented in this section were obtained assuming pressure sensors placed in nodes 12, 21, and 27. The procedure to train and test the k-NN classifier on the Hanoi network is described in the following box.

Training and testing k-NN on the Hanoi network

1. Build the pressure matrix $\mathbf{P} = [\mathbf{P}_{12}, \mathbf{P}_{21}, \mathbf{P}_{27}]$, containing one column for each sensor and one row for each leak scenario. In addition, create a vector $\mathbf{y}$ with class labels (leak nodes) containing one element for each row of $\mathbf{P}$. The matrix $\mathbf{R} = \mathbf{P} - \mathbf{P}_0$ must also be computed when pressure residuals are used as features.
2. Divide the data ($\mathbf{P}$, $\mathbf{R}$ and $\mathbf{y}$) into two groups; one for training and the other for testing. For training, the samples corresponding to leakage flow $q_\ell = 2, 4, \ldots, 50$ l/s are used, whereas for testing the samples corresponding to leakage flow $q_\ell = 1, 3, \ldots, 49$ l/s are used. In cross-validation tests, the entire data set is taken for training, and testing is performed on the validation subset in each iteration.
3. Train the k-NN classifier using $\mathbf{P}_{\text{train}}$ or $\mathbf{R}_{\text{train}}$, depending on the set of features to be used. A distance metric and a number of neighbors must be specified.
4. Test the trained classifier using the testing data $\mathbf{P}_{\text{test}}$ or $\mathbf{R}_{\text{test}}$. Predicted leak nodes are stored in a vector $\widehat{\mathbf{y}}$ to later evaluate the classifier performance.
5. Compute the classification loss (error) and the confusion matrix.

In order to find the best hyperparameters (the best combination of distance and number of neighbors) of the k-NN classifier, the classification error, as defined in (6.8), was calculated for each combination of distance and number of neighbors from $k = 1$ to $k = 50$. For each combination, a 5-fold cross-validation was applied so that the K-fold loss is used as an error metric. The K-fold loss is obtained by averaging the losses obtained on each fold. The K-fold loss for each combination of distance and number of neighbors is shown in Fig. 6.7. The definition of each distance metric was presented in Section 6.5.

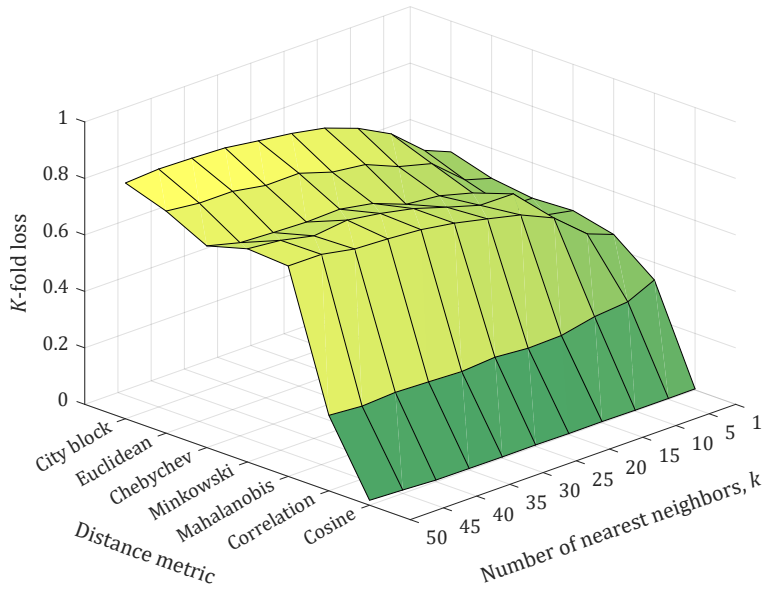


Fig. 6.7: Hyperparameter optimization in k-NN classifier for leak localization

Figure 6.7 shows that the best performance (the lowest classification error) is obtained using the cosine distance, followed by the correlation distance. Regarding the number of neighbors, the slightest classification error is obtained for 4 and 5 neighbors ($k = 4$ was selected for simplicity). Because the variation in the classification error for a different number of neighbors is barely perceptible in Fig. 6.7 for the cosine distance, this variation has been plotted separately in Fig. 6.8. This figure shows that as the number of neighbors approaches 50, the classification error increases because the data set used only contains 50 samples for each leak class.

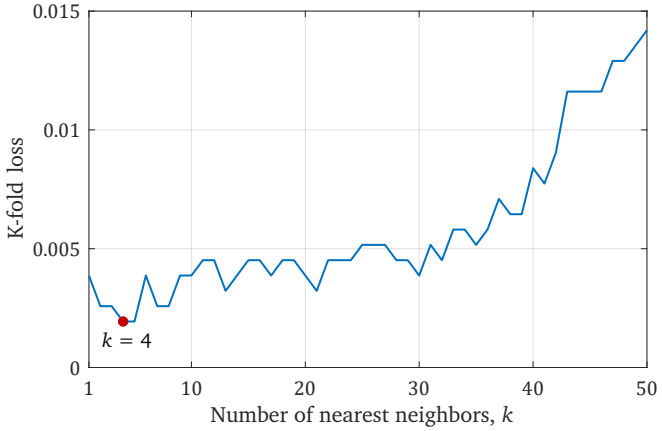


Fig. 6.8: Leak localization error using k-NN for different number of neighbors

Performance tests were applied to the k-NN classifier considering Gaussian noise in the measured pressures. The lowest classification error, obtained in noise-free conditions (SNR $= \infty$), was 0.4916 when using pressures as features, whereas it was 0.0039 when using residuals. The results plotted in Fig. 6.9 show that around 60 decibels a cut-off point is detected in the error curve where the effect of noise increases considerably. k-NN is inferior when using the measured pressures as features since almost half of the samples are wrongly classified even without noise. The use of pressure residuals (orange line) instead of raw pressures (blue line) significantly reduces leak classification error. The confusion matrices in Fig. 6.10 show an improvement in leak localization when pressure residuals are used instead of raw pressures.

From the results obtained in this section, it can be concluded that pressure residuals are an excellent option as features to classify leaks by their location. Furthermore, it was found that the best way to measure the similarity between residual pressure samples is by utilizing the cosine distance. It is also important to emphasize that when locating leaks using k-NN several neighbors greater than one must be considered according to the pressure measurements amount of noise.

6.8 Feature transformation in classifier-based leak localization

The previous section shown that the cosine distance, determined by the angle between residuals, is a good measure of similarity to classify leaks. This means that the leaks present a characteristic directionality (in the residuals space) according to the node where they occur. This section proposes a transformation on

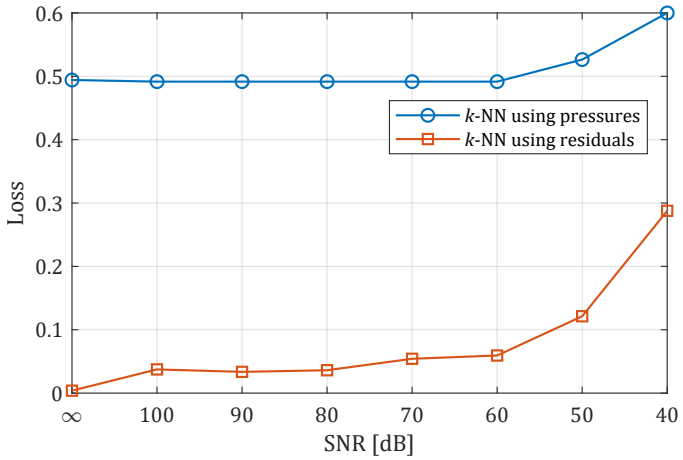


Fig. 6.9: Dependence of the classification error on the signal-to-noise ratio for k-NN with 4 nearest neighbors

the residuals to extract the information about their direction in a set of descriptors used as new features for the classifier. As shown at the end of the section, this transformation extends the concept of cosine distance used in any classifier, not just the k-NN.

According to the literature review, leak localization methods formulated as multiclass classification problems use pressure values or pressure residuals as features under a machine learning approach. An exploratory analysis of the residuals suggests that leaks at the same node tend to show a characteristic direction; see Fig. 6.11. This is a fundamental hypothesis in model-based leak localization methods based on sensitivity matrices (Pérez et al., 2011). Although in Fig. 6.11 it appears that the residuals of all the leaks at a specific node follow a constant direction, the direction of the residual vector can vary if an extensive range of leakage magnitudes is considered.

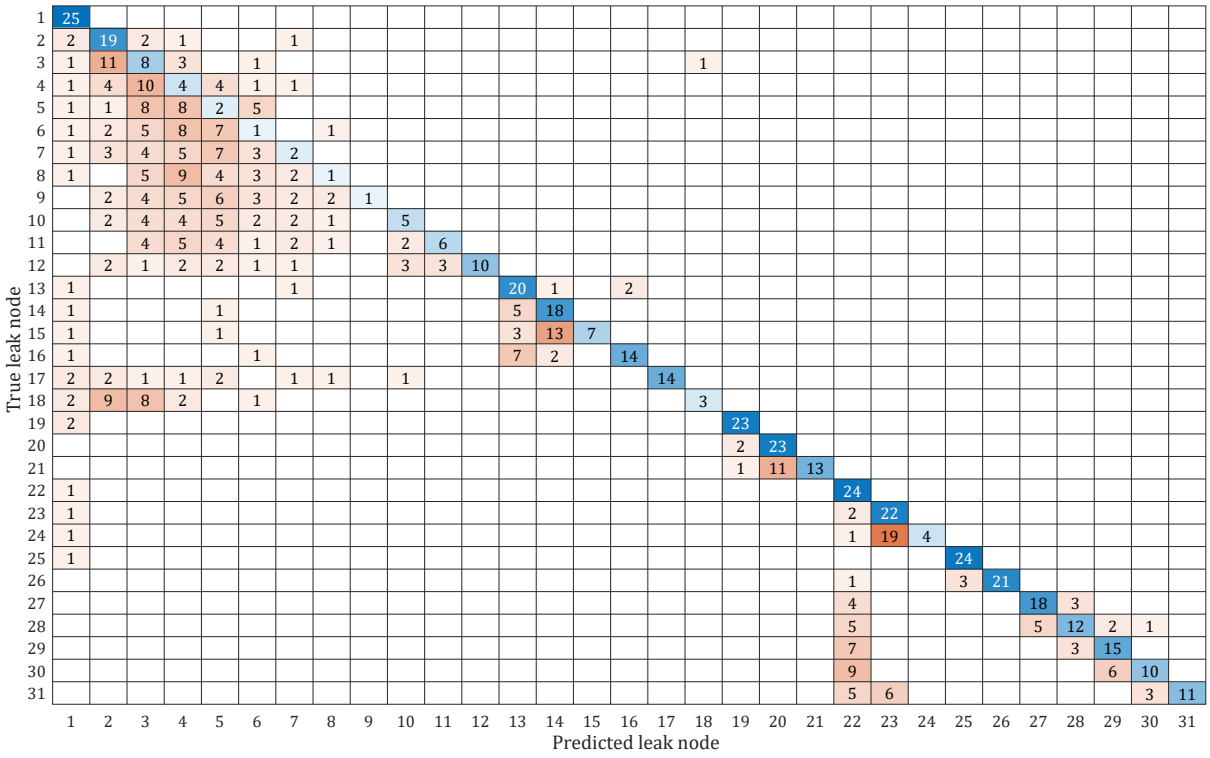
Traditional classifiers for leak localization use residual vectors in the Cartesian form where the residual magnitude and direction are implicitly combined. This does not facilitate the classification of leaks since the residual vector magnitude does not provide information on its direction. Therefore, to improve classifiers' performance for leak localization, it is proposed to map the features into a new subspace capturing only the information on the direction of the leaks and discarding the information on their magnitude.

According to vector analysis, vectors can also be expressed in a form where information about magnitude and direction is decoupled (Young, 2017). Thus, for any residual vector $\mathbf{r} = [r_1, r_2, \ldots, r_S]$, the decoupled expression is

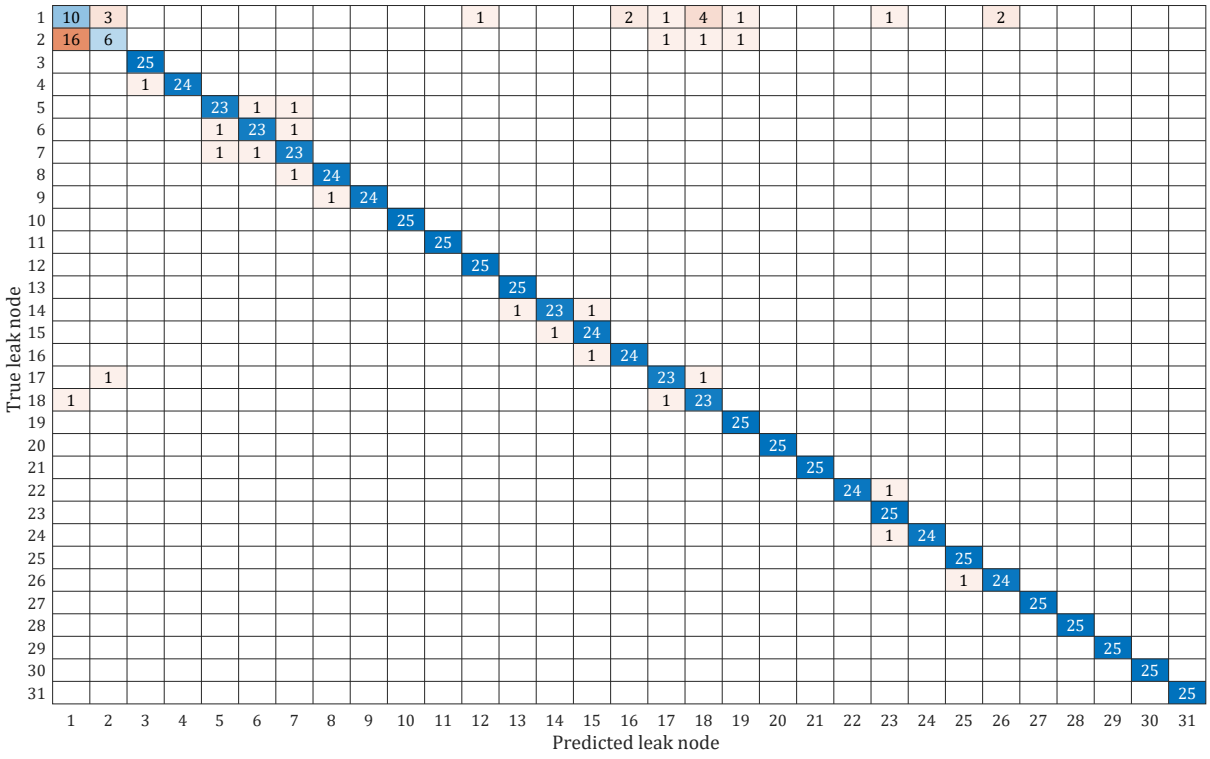
$$\mathbf{r} = M\,[\cos\theta_1, \cos\theta_2, \ldots, \cos\theta_s] = M\,[c_1, c_2, \ldots, c_S]\,, \tag{6.22}$$

where M is a scalar, and

$$c_k = f_k(r_1, r_2, \ldots, r_S) = \frac{r_k}{\sqrt{r_1^2 + r_2^2 + \cdots + r_S^2}}, \quad \text{for } k = 1, 2, \ldots, S \tag{6.23}$$



(a) Features: Pressures



(b) Features: Pressure residuals

Fig. 6.10: Confusion matrices for k-NN leak localization from noisy pressures with an SNR of 60 dB

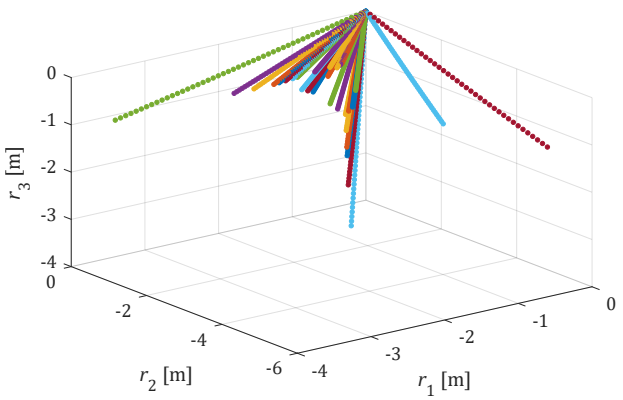


Fig. 6.11: Leaks at different nodes plotted on the residual subspace. A single color is used for each leak node, although some colors are repeated.

are the so-called **direction cosines** that uniquely describe the direction of the residual vector $\mathbf{r}$ in the S-dimensional subspace. The vector $\mathbf{c} = [c_1, c_2, \ldots, c_S]$ can be used as a new feature vector instead of the residual vector $\mathbf{r}$.

The training procedure and the predictive use of classifiers to locate leaks have been previously described by Ferrandez-Gamot et al. (2015). The modification proposed consists of using a new set of features. As shown later, replacing the original features r_k with the new features c_k improves classifiers' performance for leak localization, increasing class separability. In the diagram in Fig. 6.12, the gray box shows where the feature transformation is applied to improve the classifier's performance in the leak localization process.

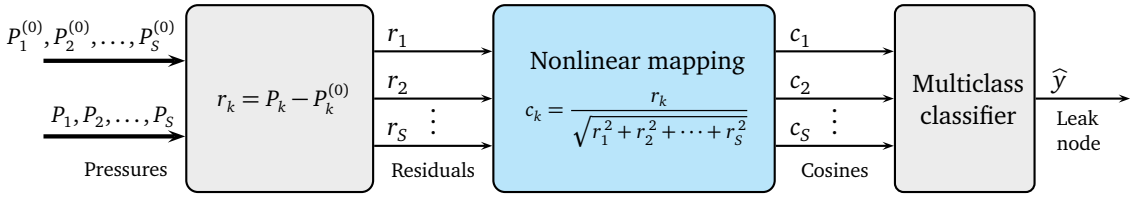


Fig. 6.12: Feature transformation to improve the classifier performance for leak localization

The proposed feature transformation can be seen as an ad hoc transformation because the original features are projected into another vector space through a nonlinear transformation. This proposal, however, does not increase the space dimension as it usually happens when the "kernel trick" (Koutroumbas and Theodoridis, 2008) is applied.

The use of direction cosines as classifiers is also related to the use of the cosine distance in k-NN. However, in k-NN, the angular proximity is measured between pairs of samples. In contrast, the direction cosines can be computed for individual samples without an additional reference sample since $[c_1, c_2, \ldots, c_S]$ measures the angular closeness of that sample concerning the implicit coordinate axes.

A series of tests verified the improvement in leak localization by using direction cosines instead of unprocessed residuals. In order to quantify the improvement obtained, the following *percentage improvement* was used

$$I\% := 100\frac{L_{\text{res}} - L_{\cos}}{L_{\text{res}}}, \tag{6.24}$$

where L_{res} is the classification loss (i.e., the leak localization error) obtained using residual features, and $L_{\cos}$ is the classification loss when using cosine features. In leak localization tests in the Hanoi network, using the data set described in Section 6.3 (sensors at nodes 12, 21, and 27), the classification loss using k-NN decreased by 99.3% when cosines were used, relative to when residuals are used directly as features. Using the naive Bayes, decision tree, and linear discriminant classifiers, the classification loss was reduced by 99.7%, 99.6%, and 97.0%, respectively, using the percentage improvement (6.24).

The lower classification loss obtained when cosines are used as features leads to better class separability. Therefore, with the four classifiers tested, the cosines better captured the directionality of the leaks in the residual subspace. This is shown in Fig. 6.13, where it is found that the class regions (where the class is detected) for three leak classes in the Hanoi network using cosine features were better defined than those for untransformed residual features. As shown in Fig. 6.13, using cosine features, the leak localization problem becomes almost independent of the classifier used since the class regions are similar for different classifiers. In fact, with cosine features, the classification loss (and consequently the accuracy) differs by less than 1% in the classifiers tested.

In order to analyze the robustness of the classifiers fed by cosine features, leak localization tests were performed considering measurement noise in node pressures. The results in Table 6.2 show that the percentage improvement when using the cosine features concerning the residual features is more significant than 50% for a wide noise margin. The percentage improvement decreases as the noise ratio in the signal increases because when the noise is considerably high (SNR < 40 decibel), the fake direction captured by the cosines become irrelevant to locate the leak. In the worst case, however, the improvement percentages remain close to 0, which means that cosine features are not decreasing in performance.

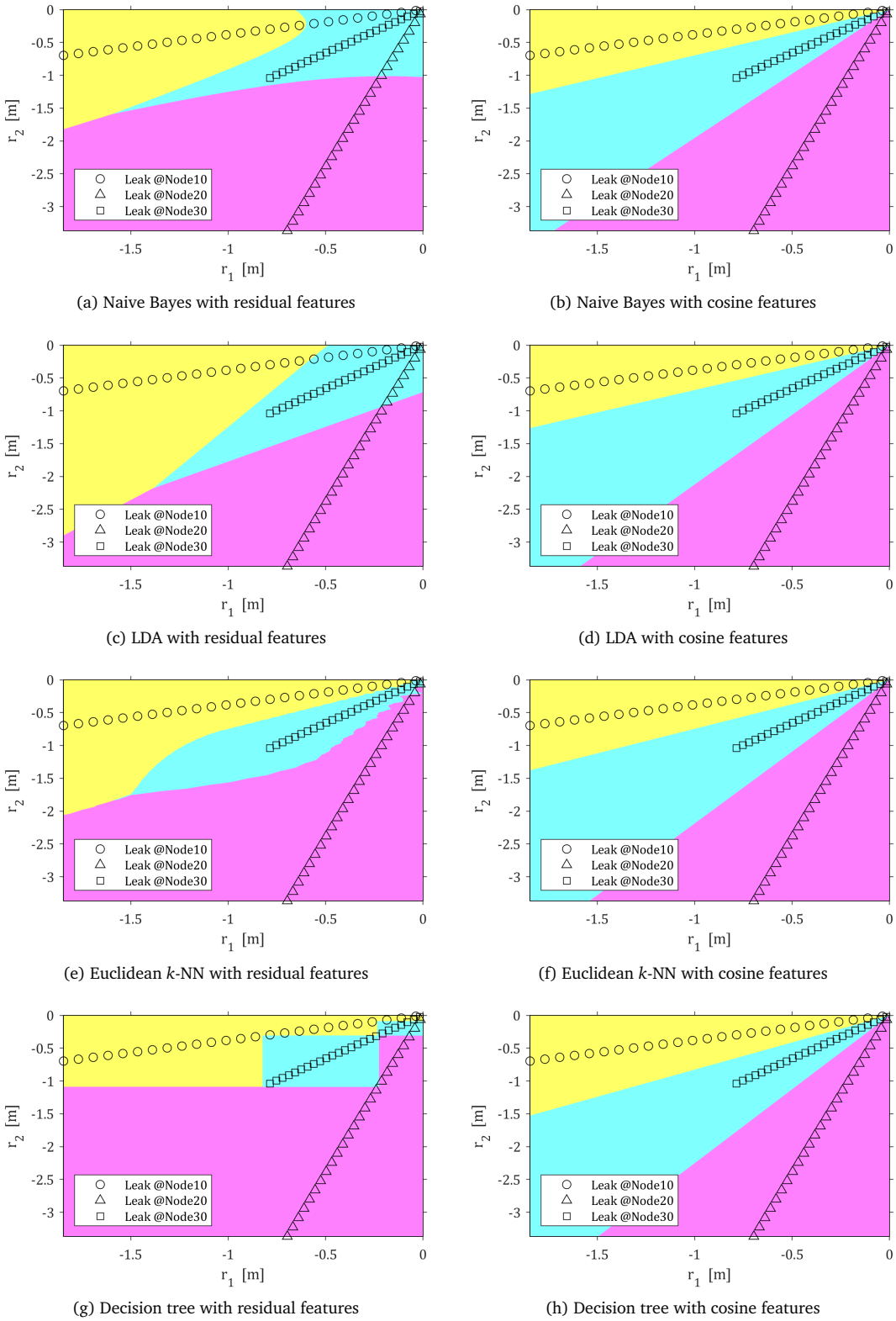
Table 6.2: Percentage improvement in performance of classifiers using cosine features

Classification method	**SNR**					
	$\infty^{\dagger}$	80 dB	60 dB	40 dB	20 dB	0 dB‡
k-Nearest Neighbors	0.993	0.873	0.394	0.006	0.013	0.005
Naive Bayes	0.997	0.939	0.405	0.009	-0.003	0.003
Decision Tree	0.996	0.917	0.473	0.021	-0.018	-0.007
Linear Discriminant Analysis	0.970	0.929	0.550	0.016	-0.040	-0.018

† Noise-free measurements

‡ Noise and signal have the same magnitude

To assess the overall performance of cosine-based classifiers in leak localization, the average topological distance (ATD) was calculated, as defined in (6.11). The results are presented on the left side of Table 6.3 for noise-free measurements, confirming the best performance when using the cosine features. By using noisy measurements, the ATD increases as the table on the right side of Table 6.3 shows, but its variation is consistent with the increase in the classification error.



(a) Naive Bayes with residual features

(b) Naive Bayes with cosine features

(c) LDA with residual features

(d) LDA with cosine features

(e) Euclidean *k*-NN with residual features

(f) Euclidean *k*-NN with cosine features

(g) Decision tree with residual features

(h) Decision tree with cosine features

Fig. 6.13: Comparison of decision boundaries for different classifiers with residual and cosine features

Table 6.3: Average topological distance using different feature sets

Noise-free measurements

Classification method	**Features**	
	Residuals	Cosines
k-Nearest Neighbors†	0.6555	0.0026
Naive Bayes	2.4090	0.0026
Decision Tree	1.3239	0.0026
Linear Discriminant Analysis	2.1974	0.0232

† Euclidean distance, $k = 5$

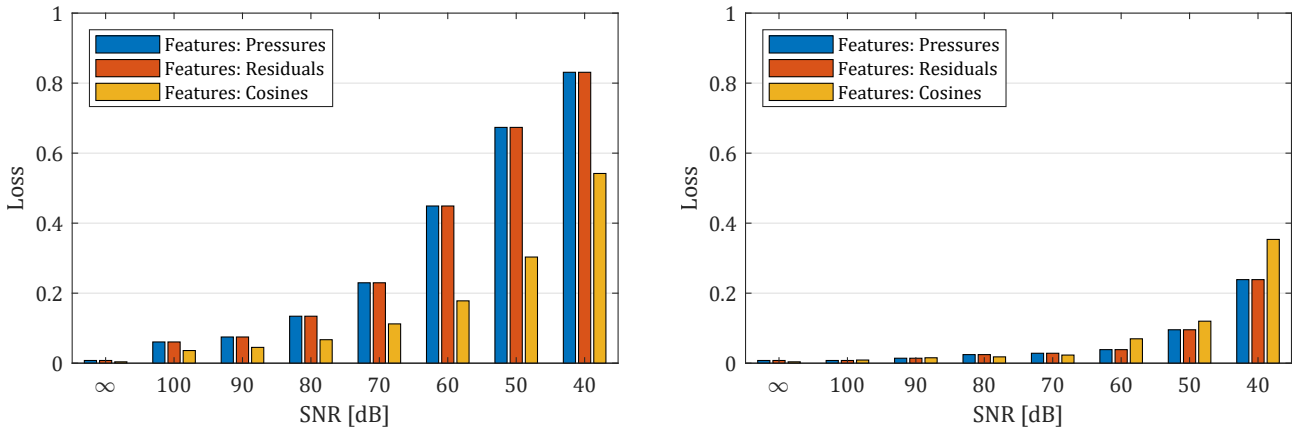
Noisy measurements, SNR = 60 dB

Classification method	**Features**	
	Residuals	Cosines
k-Nearest Neighbors†	0.9523	0.5948
Naïve Bayes	2.4387	0.9742
Decision Tree	1.3935	0.7303
Linear Discriminant Analysis	2.2090	0.7432

† Euclidean distance, $k = 5$

The quadratic discriminant analysis (QDA) classifier (not included in previous tables) deserves a separate mention in the comparative analysis of classifiers. Among the classifiers tested, this is the only one that produces good results in locating leaks using pressures without computing residuals. When residuals or cosines derived from them, however, are not used, this classifier is more sensitive to the operating point (user consumption, time of day). Therefore, its use in leak localization may require several classifiers fitted with different training sets as the operating point changes. This implies a bank of classifiers, a classifier for each hour of the day or each user consumption range.

Figure 6.14 shows the error in leak localization obtained with this classifier for different signal-to-noise ratios in the measured pressures, considering a specific time of day. This classifier's characteristic is that it showed better performance when trained from pressure measurements with added noise; see right side of Fig. 6.14. This can be explained because when the QDA classifier is trained with clean data, the class probability densities are fitted to a tiny region.



(a) Trained with noise-free data

(b) Trained with noisy data

Fig. 6.14: Classification loss in QDA-based leak localization

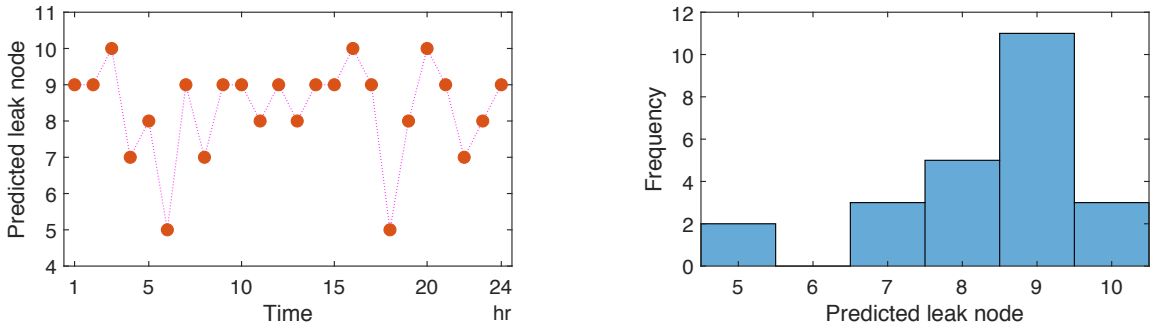
6.9 Leak localization over a time interval

Since a leak is not immediately located when it appears, a time interval T with measurements from pressure sensors is available. In this case, the features and responses are available as a time series. A refined output at a specific sample time t can be obtained, considering the class predicted during the previous interval $[t-T,\ldots,t-1,t]$. One way to refine the output is to train different classifiers for each hour of the day and predict the leak node using the one corresponding to the time of measurements. The final output (the most likely leak node) is then calculated by majority vote:

$$y_t(\mathbf{x}) = \text{mode}(\widehat{y}_\tau(\mathbf{x})), \quad \tau = t-T,\ldots,t-1,t, \tag{6.25}$$

where $\text{mode}(\cdot)$ indicates the highest frequency class. In this way, the most predicted leak node is selected over the available time interval.

Figure 6.15 shows the result of an LDA leak localization test on the Hanoi network over a full day in 1-hour increments ($t = 1,2,\ldots,24$). The true leak node is number 8, and the output obtained by majority vote is number 9, although, at different times of the day the predicted node varies within the set $\{5,7,8,9,10\}$.



(a) Predicted node over time (b) Class frequencies

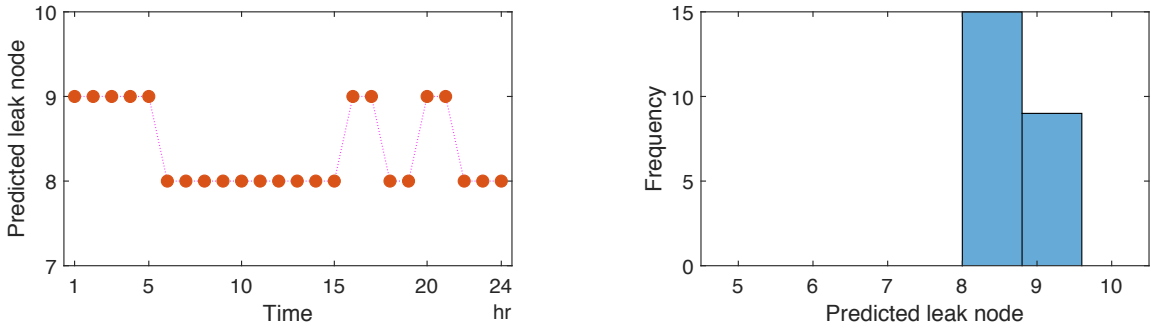
Fig. 6.15: Leak node prediction by majority vote over time

Another strategy to refine the output is, by considering the Bayesian inference described in Section 6.5, to take the posterior class probabilities at each sample time as prior probabilities for the next sample time. In this way, the prior and posterior probabilities are updated at each time step (with each new measurement) as

$$\widehat{P}_t(y_j \,|\, \mathbf{x}) = \frac{\widehat{P}_{t-1}(y_j \,|\, \mathbf{x})\, P(\mathbf{x} \,|\, y_j)}{P(\mathbf{x})}, \tag{6.26}$$

except at the initial time since there is still no posterior probability calculated. Therefore, the first prior probability is assigned as indicated in equations (6.14) and (6.15).

Figure 6.16 shows the results of the LDA leak localization test with the same test data as in Fig. 6.15 but by applying Bayesian inference over time. It can be noted that with this method the predicted node converges to the correct leak node and shows greater consistency (less variability) than the simple majority vote.



(a) Predicted node over time (b) Class frequencies

Fig. 6.16: Leak node prediction by Bayesian inference over time

It may be necessary to limit the probabilities to an interval $[0, 1-\varepsilon]$, where ε is a small number during the probability propagation over time. This prevents, when an estimated posterior probability is 1 at some sample time t_0, that the remaining probabilities are always 0 for subsequent time steps $t > t_0$.

6.10 Integrated leak localization method

The integrated procedure to locate leaks in water distribution networks, considering what has been described in previous sections, is shown in the following box. The workflow is the same for different classification methods and different feature sets.

Leak localization using classifiers

1. (**Data collection**) Using the network's hydraulic model, simulate different leakage scenarios, considering different leak flow rates and different leaky nodes. A nominal leak-free scenario should be simulated. EPANET software can be used in this step.
   From the simulation results, construct a pressure matrix, $\mathbf{P}$, containing the node pressures arranged by columns so that each row of this matrix represents a different leakage scenario. A matrix $\mathbf{P}^{(0)}$ (same size as $\mathbf{P}$) must also be constructed containing the leak-free node pressures. Additionally, a vector $\mathbf{y}$ must be constructed, which has a length equal to the number of rows in $\mathbf{P}$. This vector contains the class labels for the classifiers: the element y_i is an integer indicating the leak node for the leak scenario represented in the i-th row of $\mathbf{P}$.
2. (**Sensor placement**) From the pressure matrix $\mathbf{P}$, calculate an optimal sensor placement using the techniques proposed in Morales-González et al. (2021) and Santos-Ruiz et al. (2022). The number of sensors can be determined from a marginal analysis by progressively increasing the number of sensors until new sensors do not significantly improve leak localization performance. Some sensors less than the marginal number can be assigned depending on the available equipment.
3. (**Data partitioning**) In matrices $\mathbf{P}$ and $\mathbf{P}^{(0)}$, eliminate the columns that do not correspond to sensor nodes, keeping only the columns associated with the sensor placement calculated in the

previous step. Subsequently, separate the rows of $\mathbf{P}$ into two subsets: one for training and one for testing, creating the sub-matrices $\mathbf{P}_{\text{train}}$ and $\mathbf{P}_{\text{test}}$. The same division is performed on $\mathbf{P}^{(0)}$. Similarly, $\mathbf{y}$ is divided to obtain $\mathbf{y}_{\text{train}}$ and $\mathbf{y}_{\text{test}}$.

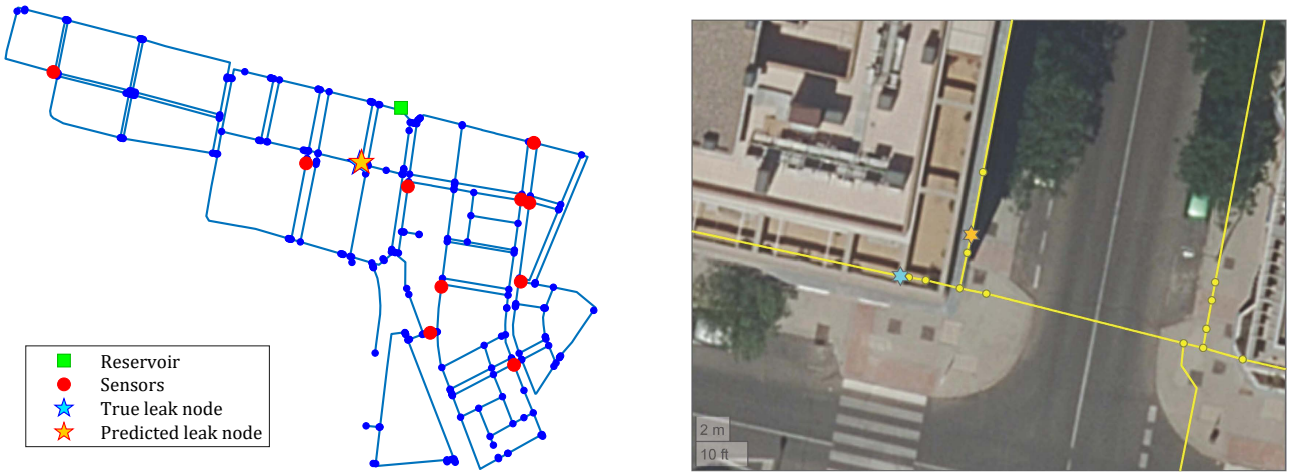
4. (**Feature computation**) Set the features for training $\mathbf{X}_{\text{train}} = \mathbf{P}_{\text{train}}$ or $\mathbf{X}_{\text{train}} = \mathbf{P}_{\text{train}} - \mathbf{P}^{(0)}_{\text{train}}$ or $\mathbf{X}_{\text{train}} = f\big(\mathbf{P}_{\text{train}} - \mathbf{P}^{(0)}_{\text{train}}\big)$, where $f(\cdot)$ is the cosine extractor function defined in (6.23). Similarly, set the features for testing: $\mathbf{X}_{\text{test}} = \mathbf{P}_{\text{test}}$ or $\mathbf{X}_{\text{test}} = \mathbf{P}_{\text{test}} - \mathbf{P}^{(0)}_{\text{test}}$ or $\mathbf{X}_{\text{test}} = f\big(\mathbf{P}_{\text{test}} - \mathbf{P}^{(0)}_{\text{test}}\big)$.
5. (**Classifier training**) Train the selected classifier using the training data labeled $\{\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}\}$.
6. (**Classifier testing**) Test the classifier using the labeled test data $\{\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}\}$. Store in a vector $\widehat{\mathbf{y}}$ the leak nodes predicted for $\mathbf{X}_{\text{test}}$. Then, evaluate the performance of the classifier from $\mathbf{y}_{\text{test}}$ and $\widehat{\mathbf{y}}$ by using the metrics described in Section 6.4.
   Steps 3 through 6 can be repeated in a cross-validation scheme. In that case, the K-fold loss is used as a performance measure.
7. (**On-line prediction**) Obtain new pressure measurements at the sensor nodes and compute the features in the same way as step 4. Use the classifier trained in step 5 to predict the leak's location. If multiple measurements are available and covering a time interval where the network's operating point changes considerably, use one of the proposals from Section 6.9 (majority vote or Bayesian inference) for the final estimate of the leak node.

The presented leak localization scheme was tested to evaluate the performance of different combinations of classifier and feature set. Table 6.4 summarizes the comparison of the performance of the different classifiers tested in the Hanoi network. In the tests, sensors were placed in three nodes (12, 21 and 27), and three different sets of features were considered: pressures, residuals and cosines. The results highlight the general improvement introduced by using cosine features instead of residuals. In addition, it is noted that k-NN with Euclidean distance and cosine features is equivalent to k-NN with cosine distance and residual features. It is also highlighted that QDA generally shows better performance than the other classifiers, except when cosine features are used, because in that case the performance of all the classifiers is quite close.

Table 6.4: Leak classification error for different classifiers on different feature sets. Error metric: 5-fold loss (cross-validation)

Classification Method	**Features**		
	Pressures	**Residuals**	**Cosines**
Naive Bayes	0.82968	0.81935	0.00645
Linear Discriminant Analysis	0.75161	0.76129	0.02000
Quadratic Discriminant Analysis	0.00645	0.00516	0.00839
Euclidean k-NN	0.48710	0.49548	0.00387
Cosine k-NN	0.67355	0.00387	0.00323
Decision Tree	0.70903	0.71161	0.00387

Using the work flow in the box above, leak localization tests were also performed in some sectors of the Madrid network (the DMA described in Section 6.3 and other medium-sized sectors) by using field measurements. An exhaustive analysis is not presented in these cases because the samples collected correspond only to a single leak point. Opening valves physically simulated the leaks. In each case, measurements were collected from 10 sensors placed in specific nodes determined by metaheuristic optimization, such as those described in Morales-González et al. (2021) and Santos-Ruiz et al. (2022). Nominal leak-free node pressures were obtained by simulation with an EPANET model tuned from measurements two days before the leak event. Figure 6.17 shows the result of the leak localization test in the DMA of the Madrid network. In this figure, the true leak node is represented by the cyan star (⋆) and the predicted leak node by the orange star (⋆). The topological distance between the true and predicted nodes (node number 272 and 138, respectively) is 5, corresponding to about 8 m of pipe length. The best leak node prediction (shown in Fig. 6.17) was obtained with k-NN using correlation distance. With k-NN using cosine distance and with other classifiers using cosine features, localization errors of about 100 m were obtained. This may be because the directionality associated with the directing cosines is sensitive both to noise in pressure measurements and to the accuracy of nominal pressure estimates obtained from the model. Therefore, poor leak-free pressure estimates can lead to fake directions associated with nodes distant from the leak node when the model is not well calibrated. In contrast, the correlation distance appears less sensitive to model calibration and measurement bias.



(a) Topological view (b) Close-up street view

Fig. 6.17: Leak localization in a DMA of the Madrid network using k-NN with correlation distance

In leak localization tests with synthetic data (simulations) in different sectors of the Madrid network, the same behavior was observed: the classifiers showed a considerably higher performance using cosine features than when they used raw residual features. In tests with physical measurements (not simulations), however, the difference was not always significant, and sometimes the k-NN with correlation distance produced better results.

6.11 Conclusions

This work has addressed a leak localization method using classifiers with transformed pressure residuals as features. The residuals' proposed transformation consists of a nonlinear mapping to extract the information about their directions, separating it from their magnitudes. The transformed features, called direction cosines, significantly improve leak localization accuracy compared with untransformed residuals, even under noisy pressure measurements. It should be noted that in predictions from real measurements in physical networks, a well-calibrated hydraulic model must be available to estimate the leak-free nominal pressures. The cosine distance and the directionality on which the cosine features are based strongly depend on the accuracy of those estimates. In contrast, the correlation distance seems less sensitive to model calibration and measurement bias, so using k-NN with correlation distance should be considered when the hydraulic model is not very accurate.

Two machine learning classification techniques were not explored in depth for leak localization and are not reported in this work: support vector machines (SVMs) and artificial neural networks (ANNs). SVMs are binary classifiers that cannot directly address the problem of classifying multiple leak classes, so their application involves using "one versus rest" or "one versus one" schemes, leading to a more complex leak localization in networks with many nodes. Furthermore, SVMs are linear classifiers, so they must be combined with kernel techniques to classify nonlinear decision boundaries. On the other hand, neural networks are a fairly broad field with an application that usually requires deep learning techniques not addressed in this work. In preliminary tests with SVMs and ANNs, no better performances were achieved than those reported with the other methods, but their exploration remains open for future work.

This page intentionally left blank

References

Akiba, T., Iwata, Y., and Yoshida, Y. (2013). Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 349–360.

Berglund, A., Areti, V. S., Brill, D., and Mahinthakumar, G. K. (2017). Successive Linear Approximation Methods for Leak Detection in Water Distribution Systems. *Journal of Water Resources Planning and Management*, 143(8):04017042.

Blocher, C., Pecci, F., and Stoianov, I. (2020). Localizing Leakage Hotspots in Water Distribution Networks via the Regularization of an Inverse Problem. *Journal of Hydraulic Engineering*, 146(4):04020025.

Box, G. (1988). Signal-to-Noise Ratios, Performance Criteria, and Transformations. *Technometrics*, 30(1):1–17.

Casillas, M. V., Garza-Castañón, L. E., and Puig, V. (2013). Extended-horizon analysis of pressure sensitivities for leak detection in water distribution networks: Application to the Barcelona network. In *2013 European Control Conference (ECC)*, pages 401–409.

Casillas, M. V., Garza-Castañón, L. E., and Puig, V. (2014). Model-based leak detection and location in water distribution networks considering an extended-horizon analysis of pressure sensitivities. *Journal of Hydroinformatics*, 16(3):649–670.

Casillas, M. V., Garza-Castañón, L. E., Puig, V., and Vargas-Martinez, A. (2015). Leak signature space: An original representation for robust leak location in water distribution networks. *Water*, 7(3):1129–1148.

De Maesschalck, R., Jouan-Rimbaud, D., and Massart, D. (2000). The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271.

Eliades, D. G., Kyriakou, M., Vrachimis, S., and Polycarpou, M. M. (2016). EPANET-MATLAB Toolkit: An Open-Source Software for Interfacing EPANET with MATLAB. In *Proc. 14th International Conference on Computing and Control for the Water Industry (CCWI)*, page 8, The Netherlands.

Ferrandez-Gamot, L., Busson, P., Blesa, J., Tornil-Sin, S., Puig, V., Duviella, E., and Soldevila, A. (2015). Leak localization in water distribution networks using pressure residuals and classifiers. *IFAC-PapersOnLine*, 48(21):220–225.

Fujiwara, O. and Khang, D. B. (1990). A two-phase decomposition method for optimal design of looped water distribution networks. *Water resources research*, 26(4):539–549.

Koutroumbas, K. and Theodoridis, S. (2008). *Pattern recognition*. Academic Press.

Martinez, W. L. and Martinez, A. R. (2015). *Computational Statistics Handbook with MATLAB*. Computer Science and Data Analysis Series. Chapman and Hall/CRC, 3 edition.

Morales-González, I., Santos-Ruiz, I., López-Estrada, F., and Puig, V. (2021). Pressure Sensor Placement for Leak Localization Using Simulated Annealing with Hyperparameter Optimization. In *2021 5th International Conference on Control and Fault-Tolerant Systems (SysTol)*, pages 205–210. IEEE.

OECD (2016). Water Governance in Cities. *OECD Studies on Water*.

Pérez, R., Puig, V., Pascual, J., Peralta, A., Landeros, E., and Jordanas, L. (2009). Pressure sensor distribution for leak detection in Barcelona water distribution network. *Water Science and Technology: Water Supply*, 9(6):715–721.

Pérez, R., Puig, V., Pascual, J., Quevedo, J., Landeros, E., and Peralta, A. (2011). Methodology for leakage isolation using pressure sensitivity analysis in water distribution networks. *Control Engineering Practice*, 19(10):1157–1167.

Pérez, R., Quevedo, J., Puig, V., Nejjari, F., Cugueró, M. A., Sanz, G., and Mirats, J. M. (2011). Leakage isolation in water distribution networks: A comparative study of two methodologies on a real case study. In *2011 19th Mediterranean Conference on Control Automation (MED)*, pages 138–143.

Pérez, R., Sanz, G., Puig, V., Quevedo, J., Cugueró, M. À., Nejjari, F., Meseguer, J., Cembrano, G., Mirats, J. M., and Sarrate, R. (2014). Leak localization in water networks: A model-based methodology using pressure sensors applied to a real network in Barcelona [applications of control]. *IEEE Control Systems Magazine*, 34(4):24–36.

Pilcher, R., Hamilton, S., Chapman, H., Ristovski, B., and Strapely, S. (2007). Leak Location and Repair Guidance Notes. *International Water Association—Water Loss Task Forces: Specialist Group Efficient Operation and Management, Bucharest, Romania*.

Pudar, R. S. and Liggett, J. A. (1992). Leaks in Pipe Networks. *Journal of Hydraulic Engineering*, 118(7):1031–1046.

Puig, V., Ocampo-Martínez, C., Pérez, R., Cembrano, G., Quevedo, J., and Escobet, T., editors (2017). *Real-time Monitoring and Operational Control of Drinking-Water Systems*. Springer International Publishing.

Rossman, L. A., Woo, H., Tryby, M., Shang, F., Janke, R., and Haxton, T. (2020). EPANET 2.2 User Manual. Technical Report EPA/600/R-20/133, U.S. Environmental Protection Agency, Washington, DC.

Santos-Ruiz, I., López-Estrada, F.-R., Puig, V., Valencia-Palomo, G., and Hernández, H.-R. (2022). Pressure Sensor Placement for Leak Localization in Water Distribution Networks Using Information Theory. *Sensors*, 22(2).

Sanz, G. and Pérez, R. (2015). Sensitivity Analysis for Sampling Design and Demand Calibration in Water Distribution Networks Using the Singular Value Decomposition. *Journal of Water Resources Planning and Management*, 141(10):04015020.

Sanz, G., Pérez, R., Kapelan, Z., and Savic, D. (2016). Leak Detection and Localization through Demand Components Calibration. *Journal of Water Resources Planning and Management*, 142(2):04015057.

Young, E. C. (2017). *Vector and Tensor Analysis*. Pure and Applied Mathematics. CRC Press, 2 edition.

Chapter 7
LabVIEW-Based SCADA System for Sequential Leaks' Diagnosis in Pipelines

Rolando Carrera[1] and Cristina Verde[1]

Most of the leak diagnosis contributions have only been tested off-line by the academic community, and there are few studies on the implementation of algorithms as an integral part of a SCADA (Supervisory Control And Data Acquisition) system with diagnostic capability. This paper aims to fill the lack of valuable knowledge from a practical point of view about the implementation of diagnosis systems. Specifically, we discuss one that introduces the design of LabVIEW software implemented on a CompactRIO controller, tasked with localizing sequential leaks according to an algorithm previously developed at the Instituto de Ingeniería, UNAM. This program was designed as part of the SCADA system for a hydraulic pilot plant. Some recommendations and remarks are addressed based on the authors' experience.

7.1 Introduction

Fault detection and identification (FDI) is a relative new research area in the automatic control community. FDI solutions have been developed almost for every engineering field. One field of interest is faults in pipelines because they may cause heavy financial losses or worst: environment damages or human casualties (Fang et al., 2019). Thus, for this purpose different pipeline leakage detection methods have been performed. For instance: exterior methods that are based on sensors that detect the presence of the conducted material in the environment along the pipeline; interior or computational methods based on dynamic behavior of the fluid and; visual or biological methods based on human or artificial vision or acute smell capacity of dogs (Fang et al., 2019).

Diverse technologies for leak diagnosis through software have been developed based on the model of the dynamics of a liquid element in a pipe, starting with the works of Billman and Isermann (1987) by assuming only available data of pressure and flow rate at the ends of the pipeline. Recently, the state of the art of the leak diagnosis methods has been published in Moubayed et al. (2021), Zhang et al. (2015) and Liu et al. (2019). The detectors by software use data driven models such as neural networks, analytical models derived from the dynamic model, and also combinations of both models. Thus, one can find simple static relationships easy to implement (Carrera and Verde, 2001) up to complex observers for hyperbolic equations by boundary injection (Aamo, O. M and Salvesen, J and Foss, 2006).

[1]Instituto de Ingeniería, UNAM, Mexico, e-mail: rcarrera@unam.mx and verdeunam.mx

In contrast to the diverse options that isolate a single leak, few methods have been validated by considering a multiple leak scenario. This fact could be motivated by the remark about the weak detectability of two simultaneous leaks published in Verde et al. (2003) and the practical condition of avoiding permanent manoeuvers in a line for inflammable fluids. These issues imply that some leaks cannot be isolated, once the transient response produced by the leak has disappeared. On the other hand, when the useful lifetime of a pipeline is exceeded, sequential leaks could occur. This fact motivated the search for possible solutions for the multiple sequential leaks (*MSL*) problem in which the time interval required by the application to isolate the leaks determines how close in time the leaks can appear.

One research line in the Coordinación de Eléctrica y Computación at the Instituto de Ingeniería, UNAM, is fault detection and identification of complex systems, specifically early fault detection in pipelines with this goal in mind, a pipeline prototype (Carrera, 2019) was built in the Hydromechanics Laboratory, as seen in Fig. 7.1. Data from experiments can be downloaded in order to perform FDI off line. More importantly, however, with this facility one can test FDI methods in real-time.

Fig. 7.1: Partial view of the facility. Pipeline in blue to the right and hydraulic pumps, instruments and water reservoir to the left

The prototype is instrumented with flow rate and pressure sensors and is also complemented with a data acquisition system and a personal computer for software issues. This prototype has been used to test several fault detection methods throughout its existence (the first prototype was built in 1999), currently we are testing wave pressure and multiple leaks detection methods. The last method is the focus of this chapter.

In this chapter we present a new supervisory control and data acquisition (SCADA) system called SCADA-II (Instituto de Ingeniería). The goal of this system is to apply in real-time a multiple sequential leaks method in our prototype. In the following sections, the whole process of achieving this goal is explained.

7.2 Experimental Prototype Description

The prototype is a pipe without lateral branches with coil geometry, with pressure and flow sensors at the ends, and several valves and servo valves for emulation of leaks along the pipe. Pressure sensors can be attached at the ends of each level of the coil. Therefore, this prototype is suitable for the application of the method described in Section 7.3.

7.2.1 Prototype configuration

The fault detection prototype was built with a 7.62 mm diameter iron pipe. Table 7.1 shows its physical parameters, and the layout of the pipe is shown in Fig. 7.2. The main parts of the prototype are the following: a) 10 m^3 water reservoir; b) two hydraulic pumps of 7.5 and 10 HP, which can be combined or used alone; c) Coil shape pipe on a vertical plane that ends in the drinking water reservoir; d) ball intake valves for flow regulation; e) pressure and flow measure points; and f) valves for leaks emulation. The downstream pressure transducer is located 4.5 m from the end of the pipe, which is open to the atmospheric pressure. The prototype is monitored and regulated by a personal computer (PC) attached to a data acquisition system, which monitors the variables of flow rate and pressure, actuates the servo valves and modifies the hydraulic pump velocity. The personal computer, the data acquisition system and the program in the PC compose a SCADA system.

Table 7.1: Pilot pipeline parameters

Parameters	Values	Description
D	$0.076\,\mathrm{m}$	Inner diameter
L	$163.715\,\mathrm{m}$	Length
g	$9.81\,\mathrm{m/s^2}$	Gravity acceleration
a	$1330\,\mathrm{m/s}$	Pressure wave velocity
ν	10^{-6} $[\mathrm{m^2/s}]$ at 20^o C	Fluid kinematic viscosity
ε	0.2 mm	Rugosity
Material		Schedule 40 galvanized iron

The mathematical model that describes the flow behavior in a pipe was developed from the analysis of a straight pipe. Because of space limitations, however, the prototype is an array of pipes in a coil shape on a vertical plane. This array is not a drawback since from this array one can have an equivalent in the shape of a straight horizontal pipe by using the height relative to a reference level (Section 7.2.2).

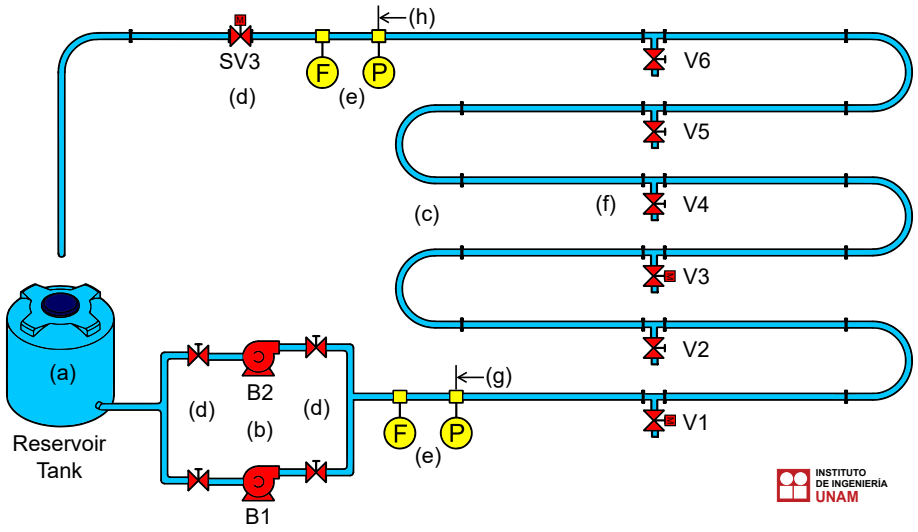


Fig. 7.2: Schematic of the prototype: a) water reservoir, b) hydraulic pumps, c) pipe, d) input and output valves, e) flow and pressure sensors, f) leak valves, g) beginning of the pipe and h) end point

The part of the pipe useful for fault detection is the coil between measurement instruments. These two points, (g) and (h) (see Fig. 7.2), are the marks of the beginning and ending of the monitored pipe. There are also six points where leaks can be created; these points are composed of 2.54 cm diameter iron pipes welded to the main pipe and ball valves that regulate the leak flow (valves V1 to V6, Fig. 7.2). Two of these valves are servo valves (V1 and V3) that are manipulated in the SCADA-II in order to synchronize time and set the size of the leak.

7.2.2 Topographic profile and pressure modification

The prototype has the layout shown in Fig. 7.2. As the area and conduit material are constant along the pipe, then the pressure head (Bansal, 2005) is

$$\tilde{H}(z) = \frac{P(z)}{\rho g}, \tag{7.1}$$

where $P(z)$ is the manometric pressure at point z, ρ is the fluid density and g the gravity constant. Furthermore, (7.1) can be written in terms of piezometric head $H(z) = \tilde{H}(z) + h(z)$ where $h(z)$ is the pipe height over any level reference, and z is the spatial coordinate along the pipe.

To handle this pilot plant as it were a horizontal plane conduit, the whole length L is calculated by adding the length of the straight and u-shape sections, and by taking the pressure as the water column height that is achieved by transforming the pressure head $\tilde{H}(z)$ according to the reference level height $h(z)$ with $h(0) = 0\,m$, at the inlet pipe (Sieber and Isermann, 1977). The last assumption is the same as adding gravity pressure (caused by the height difference to the reference level) to the pressure head at any point z of the line. Experimental studies have shown that for the specific parameters of the pilot plant the model error produced by the above assumptions is neglected (Carrera and Verde, 2001).

7.2.3 Sensors and equipment

In this subsection, the main features of the sensors are described, as well as the equipment used in the facility: hydraulic pumps, electronic power inverters that drive the pumps, servo valves, data acquisition system (DAQ) and personal computer (PC).

Flow sensors: Promass 83F from Endress + Hauser (E+H, 2018b), sensor based on the Coriolis effect, $\pm 0.05\%$ maximum error

Pressure sensors: Cerabar PMO71 from Endress + Hauser (E+H, 2018a), piezoresistive sensor, $\pm 0.05\%$ maximum error

Inverters: Frequency variators F800 and E740 from Mitsubishi (Mitsubishi, 2018)

Centrifugal pumps: Siemens 10 HP and 7.5 HP (Barmesa, 2018)

Data interface: compactRIO (NI, 2018) with A/D and D/A converters

Servo valves: Georg Fischer type 133 (GF, 2018), and Remote Control RCE 15, (RC, 2018)

Personal computer: Panasonic Toughbook CF-53 (Panasonic, 2013), 64-bit processor, 2.5 GHz clock, 446 GB HD and 4 GB RAM

All sensors, servo valves and inverters are connected to the data interface by means of 4-20 mA current loops. The PC communicates with the data interface via an USB connector. More details are given in Subsection 7.4.2.

7.3 Diagnosis Model for Two Sequential Leaks

The transient flow model in pressurized pipes is based on the conservation laws of momentum and mass that govern transient flows. The partial differential equations:

$$\begin{aligned} \frac{1}{a_1}\frac{\partial Q(z,t)}{\partial t} + \frac{\partial H(z,t)}{\partial z} + J(Q(z,t),\theta_1) = 0, \\ \frac{\partial H(z,t)}{\partial t} + a_2\frac{\partial Q(z,t)}{\partial z} = 0 \end{aligned} \tag{7.2}$$

describe a fluid in a one dimensional straight line of one-dimension in the absence of leaks and are taken from Chaudhry (2014), where $H(z,t)$ and $Q(z,t)$, respectively, denote the piezometric pressure head and the flow rate at the center of the pipe for a pipe of length L. Furthermore, $z \in [0,L]$ is the spatial variable, and $t \geq 0$ is the time variable. The parameters are $a_2 = a^2/a_1$ with a the pressure wave speed, $a_1 = gA_r$ with A_r the constant cross-sectional area and g is the gravitational acceleration. The hydraulic gradient function per unit length, $J(Q(z,t),\theta_1) = \theta_1 Q^2$, depends on specific friction approximations, and here a quadratic function is used, where θ_1 is the parameter associated with the pressure gradient. Model (7.2) is developed by assuming elastic pipes and weakly compressible fluids with a flow rate significantly lower than the pressure wave speed. This model is the base of diverse leak diagnosis methods by using mass balance and the hydraulic gradient of the fluid. One can consult the reference Verde and Torres (2017), where diverse methods are introduced.

A leak at point L_e on the pipe is characterized by:

$$\begin{aligned} Q(L_e-\varepsilon,t) =& Q(L_e+\varepsilon,t)+\mu(t-\tau)Q_e(t), \\ H(L_e-\varepsilon,t) \approx& H(L_e+\varepsilon,t) \end{aligned} \tag{7.3}$$

with ε an infinitesimal value close to 0, and the time τ defines the initialization of the leak with μ as the step function. The outflow rate induced by the piezometric head at L_e is given by:

$$Q_e(t) = \lambda_e \sqrt{H_e(L_e,t)}, \tag{7.4}$$

where the leak coefficient λ_e depends on the discharge coefficient, the leak cross section area s_a and the gravitational acceleration g. Thus, the leak conditions (7.3) and (7.4) must be considered at each point where a leak is present in the pipeline.

To evaluate the SCADA-II system, which is the subject of this chapter, a MSL (Multiple Sequential Leaks) method developed at the Instituto de Ingeniería has been integrated to the data acquisition system. From an implementation point of view, the main advantages of the method are

- the high velocity of the location of the leak since the dimension of the estimation algorithms is of minimum order and independent of the number of leaks.
- the continuous identification of the gradient parameter simultaneously with the leaks estimation during the monitoring task.

The reader interested in the detail of the algorithm derivation can consult Rojas (2021). Here, only the main details are given. A key result to simplify the MSL problem is the set of equations obtained from Eqs. 7.2, 7.3 and 7.4, which characterizes the models equivalence, in steady state, when there are two leaks and one leak, with the friction as a function of the flow rate by sections (Verde and Rojas, 2017). Figure 7.3 describes variables and parameters of the pipelines used to characterize the equivalence relationships between one and two leaks using an input-output model.

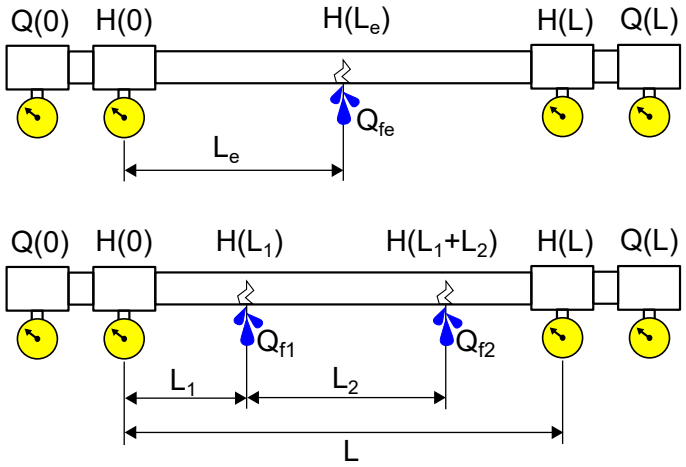


Fig. 7.3: Variables description of the equivalent models in steady state for the pipeline with one leak in $z_e = L_e$ and with two leaks ($z_1 = L_1$ and $z_2 = L_1 + L_2$)

Fact 1: Let the two leaks scenarios be in steady state as shown in Fig. 7.3 where both pipelines have the same physical characteristics (length, diameter, fluid and roughness). Assume that the outflow Q_e is

Table 7.2: Fluid Models by Sections with one and two Leaks

Leak L_e	**Leaks L_1 y L_2**
Upstream Section	
$z_{1e} \in [0, L_e]$	$z_1 \in [0, L_1]$
$\frac{1}{a_1}\frac{\partial Q(z_{1e},t)}{\partial t} + \frac{\partial H(z_{1e},t)}{\partial z_{1e}} + J\big(Q(z_{1e}(t),\theta_1)\big) = 0$ $\frac{\partial H(z_{1e},t)}{\partial t} + a_2\frac{\partial Q(z_{1e},t)}{\partial z_{1e}} = 0$	$\frac{1}{a_1}\frac{\partial Q(z_1,t)}{\partial t} + \frac{\partial H(z_1,t)}{\partial z_1} + J\big(Q(z_1,t),\theta_1)\big) = 0$ $\frac{\partial H(z_1,t)}{\partial t} + a_2\frac{\partial Q_0(z_1,t)}{\partial z_1} = 0$
Boundary Condition	Boundary Condition
$H(0,t)$ $Q(L_e-\varepsilon,t) = Q(L_e+\varepsilon,t) + Q_e(t)$ $H(L_e-\varepsilon,t) = H_L(L_e+\varepsilon,t)$	$H(0,t)$ $Q(L_1-\varepsilon,t) = Q(L_1+\varepsilon,t) + Q_1(t)$ $H(L_1-\varepsilon,t) = H(L_1+\varepsilon,t)$
Middle Section	
-	$z_2 \in [L_1, L_1+L_2]$
-	$\frac{1}{a_1}\frac{\partial Q(z_2,t)}{\partial t} + \frac{\partial H(z_2,t)}{\partial z_2} + J\big(Q(z_2,t),\theta_1)\big) = 0$
-	$\frac{\partial H(z_2,t)}{\partial t} + a_2\frac{\partial Q(z_2,t)}{\partial z_2} = 0$
-	Boundary Condition
-	$Q(L_1+L_2-\varepsilon,t) = Q(L_1+L_2+\varepsilon,t) + Q_2(t)$
-	$H(L_1+L_2-\varepsilon,t) = H(L_1+L_2+\varepsilon,t)$
Down Stream Section	
$z_{2e} \in [L_e, L]$	$z_3 \in [L_1+L_2, L]$
$\frac{1}{a_1}\frac{\partial Q(z_{2e},t)}{\partial t} + \frac{\partial H(z_{2e},t)}{\partial z_{2e}} + J\big(Q(z_{2e},t),\theta_1)\big) = 0$ $\frac{\partial H(z_{2e},t)}{\partial t} + a_2\frac{\partial Q(z_{2e},t)}{\partial z_{2e}} = 0$	$\frac{1}{a_1}\frac{\partial Q(z_3,t)}{\partial t} + \frac{\partial H(z_3,t)}{\partial z_3} + J\big(Q(z_3,t),\theta_1)\big) = 0$ $\frac{\partial H(z_3,t)}{\partial t} + a_2\frac{\partial Q(z_3,t)}{\partial z_3} = 0$
Boundary Condition $H(L,t)$	Boundary Condition $H(L,t)$

related to the outflow rates of line 2 by

$$Q_e = Q_{f1} + Q_{f2} \tag{7.5}$$

with Q_{f1} and Q_{f2} as the outflows of pipeline 2, respectively. Thus, the differences between both lines are described next:

Pipeline 1. A leak with flow rate Q_e at L_e in line 1, and
Pipeline 2. Two leaks with flow rates Q_{f1} and Q_{f2} at coordinates L_1 y L_1+L_2, respectively

Considering the segments between leaks of Fig. 7.3, each fluid can be described by equations of the form (7.2) and its respective boundary conditions. Thus, the models shown in Table 7.2 were obtained for each section.

If both flows are in equilibrium conditions, from the models of Table 7.2, the relation between the leaks positions and the hydraulic gradients of the three sections is given by

$$\Big(L_e - L_1\Big)\Big(J(Q(0),\theta_1) - J(Q(L),\theta_1)\Big) = L_2\Big(J((Q(0)-Q_{f1}),\theta_1) - J(Q(L),\theta_1)\Big) \tag{7.6}$$

for the two pipelines are obtained. This equation is an input-output equivalent model for two and one leak and can be interpreted as pressure indistinguishable profiles for both lines at their ends, respectively. These indistinguishability regions justify the necessity of a transitory regime to locate simultaneous leaks. In terms of the parameter λ_1 and the piezometric pressure head at the ends of the pipeline, the distance between the first and second leak can be written as

$$L_2 = \frac{H(0) - H(L) - L_1 J\big(Q(0),\theta_1\big) - (L-L_1)J\big(Q(L),\theta_1\big)}{\theta_1\Big(Q(0) - \lambda_1\Big[H(0) - L_1 J\big(Q(0),\theta_1\big)\Big]^{1/2}\Big)^2 - J\big(Q(L),\theta_1\big)}. \tag{7.7}$$

Moreover, by considering (7.5), the parameters associated with the second leak can be expressed as

$$\lambda_2 = \frac{Q(0) - Q(L) - \lambda_1\Big(H(0) - L_1 J\big(Q(0),\theta_1\big)\Big)^{1/2}}{\Big(H(L) + (L - L_1 - L_2)J\big(Q(L),\theta_1\big)\Big)^{1/2}}. \tag{7.8}$$

Finally, from the hydraulic gradient along both lines the inequality

$$z_1 = L_1 < L_e < z_2 = L_1 + L_2 \tag{7.9}$$

can be validated. In other words, the positions of the leaks in a two leaks scenario are upstream and downstream of the equivalent leak position L_e.

Thus, if the pair (L_1, λ_1) together with the equivalent leak position L_e are assumed to be previously known, one can estimate the rest of the unknown variables of the two leaks pipeline from (7.6), (7.7), (7.8) and the condition imposed by the inequality (7.9). This means L_2, λ_2 and Q_{f2} can be estimated.

Therefore, by considering the above relationships and that only one leak occurs at the same time, an iterative algorithm can be implemented as part of the monitoring system. The algorithm will be activated each time a new equivalent appears and the equivalent leak (L_e, λ_e) is estimated. The values of this estimation together with the historical leakage data of the line given by L_i and λ_i then allow the estimation of the values associated with the leak $i+1$. The following subsection addresses a proposed solution for the estimation of sequential leaks with small occurrence times between them.

7.3.1 Sequential leak location algorithm

By assuming that the physical parameters of the duct are known and the data of the flow rates and piezometric pressures at the ends of the line $(Q(0), Q(L), H(0), H(L))$ are available, the algorithm for the location of two sequential leaks on-line is described here. Since the friction parameter uncertainty has been recognized in experimental studies (Billman and Isermann, 1987; Rojas et al., 2021), the diagnostic algorithm is robustified by including an adaptive observer for the identification on-line of the gradient

$J(Q,\theta_1)$. Moreover, to reduce the number of false leaks, an innovative redundant approach for symptom generation is also incorporated. The equivalent model

$$\begin{aligned}\dot{Q}(0) &= \frac{a_1}{L_e}\Big(H(0)-H(L_e)\Big)-a_1 J(Q(0),\theta_1),\\ \dot{H}(L_e) &= \frac{a_2}{L_e}\Big(Q(0)-Q(L)-Q_e(t)\Big),\\ \dot{Q}(L) &= \frac{a_1}{L-L_e}\Big(H(L_e)-H(L)\Big)-a_1 J(Q(L),\theta_1)\end{aligned} \tag{7.10}$$

is used for the identification of the pair (Q_e,L_e) with $L>L_e>0$, or of the parameter θ_1, according to the assigned task.

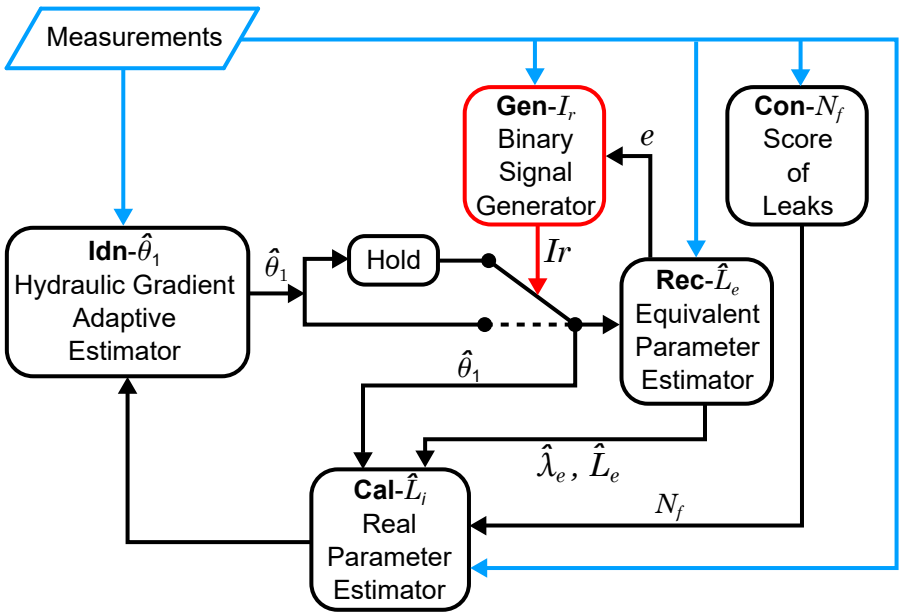


Fig. 7.4: Diagnosis system scheme with main tasks $\mathbf{Gen-I_r}$, $\mathbf{Rec-\hat{L}_e}$, $\mathbf{Idn-\theta_1}$, $\mathbf{Cal-\hat{L}_i}$, $\mathbf{Con-N_f}$, and the holder and the switch

The algorithm consists mainly of coupled estimation tasks driven by the symptom or binary signal

$$I_r = \begin{cases} 0 \text{ if the system is awaiting the next leak} \\ 1 \text{ if the leak parameters are being estimated,}\end{cases}$$

as a function of the data set and residual state. Figure 7.4 shows the interconnection of the estimators where the prescribed tasks are detailed below.

- $\mathbf{Gen-I_r}$. Generation of the binary signal I_r that governs the state of the algorithm according to the following parallel conditions of the flow rates
  - Condition 1 is active, if the convergence error given by $S_{EKF}=|e_1|+|e_2|$ of the EKF (Extended Kalman Filter) with $e_1=Q(0)-\hat{Q}(0)$ and $e_2=Q(L)-\hat{Q}(L)$ is greater than the positive threshold T_{EKF}.

– Condition 2 is active, if the difference of the flow rates at the ends $r = Q(0) - Q(L)$ is greater than the positive threshold th.

Thus, if both conditions are not satisfied, I_r is off and $\hat{\theta}_1$ is calculated over and over in (7.10). In addition, the EKF simultaneously generates e_1 and e_2. If both conditions are active, $I_r = 1$, and the value of $\hat{\theta}_1$ is held. Furthermore, the EKF identifies the new equivalent parameters L_e and λ_e. Both thresholds of the above conditions are adjusted according to Isermann (2006).

- **Idn** $-\hat{\theta}_1$. Identification of $\hat{\theta}_1$ with known L_e. This task is feasible according to Besançon (2007) by assuming unknown signals $H(L_e)$ and θ_1. Defining the output $y = \begin{pmatrix} Q(0) & Q(L) \end{pmatrix}^T$, the input $u = \begin{pmatrix} H(0) & H(L) \end{pmatrix}^T$, and the unknown constant vector $\Theta = \begin{pmatrix} \theta_1 & H(L_e) \end{pmatrix}^T$ the adaptive observer assumes the model (7.10) transformed to the canonical observer form

$$\dot{y} = \alpha(y,u,t) + \beta(y,t)\Theta \tag{7.11}$$

with $\alpha(y,u,t) = \begin{pmatrix} \frac{a_1}{L_e}u_1 \\ -\frac{a_1}{L-L_e}u_2 \end{pmatrix}$ and $\beta(y,t) = \begin{pmatrix} -a_1 y_1^2 & -\frac{a_1}{L_e} \\ -a_1 y_2^2 & \frac{a_1}{L-L_e} \end{pmatrix}$. Thus, according to Besançon (2007), the adaptive system

$$\begin{aligned} \dot{\hat{y}} &= \alpha(\hat{y},u,t) + \beta(\hat{y},u,t)\hat{\Theta} - k_y(\hat{y}-y); \text{ for any scalar } k_y > 0 \\ \dot{\hat{\Theta}} &= -k_\theta \beta(\hat{y},u,t)^T(\hat{y}-y); \text{ for any scalar } k_\Theta > 0, \end{aligned} \tag{7.12}$$

which includes Θ as a new state, is an observer of (7.11). Furthermore, if β is persistently exciting (Narendra and Annaswamy, 1987) and $\dot{\beta}$ is bounded, the parameter error also goes to 0. Thus, the observer (7.12) is implemented for the estimation of θ_1.

- **Rec** $-\hat{\mathbf{L}}_\mathbf{e}$. Reconstruction of the equivalent parameters L_e and λ_e for the equivalent system (7.10) with a known constant value of $\hat{\theta}_1$ by applying an EKF. The EKF is designed by considering that the dynamics of $H(L_e)$ is faster than the dynamics of $Q(0)$ and $Q(L)$. This assumption allows the replacement of the second state in (7.10) by $\dot{H}(L_e) = 0$ with the constraint

$$0 = \hat{Q}(0) - \hat{Q}(L) - \hat{\lambda}_e\sqrt{\hat{H}(\hat{L}_e)}. \tag{7.13}$$

Therefore, the identification task can be solved with the augmented model

$$\begin{aligned} \dot{x} &= f(x,u,\hat{\theta}_1), \\ y &= Cx \end{aligned} \tag{7.14}$$

with

$$x = \begin{pmatrix} Q(0) & Q(L) & H(L_e) & L_e \end{pmatrix}^T,\ C = \begin{pmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \end{pmatrix} \text{ and}$$

$$f(x,u,\hat{\theta}_1) = \begin{pmatrix} \frac{a_1}{x_4}(u_1 - x_3) - a_1 J_s(x_1,\hat{\theta}_1) \\ \frac{a_1}{L-x_4}(x_3 - u_2) - a_1 J_s(x_2,\hat{\theta}_1) \\ 0 \\ 0 \end{pmatrix},$$

where only the unknown constant parameter L_e is added to (7.10). Therefore, according to Reif et al. (1998), the EKF algorithm for the continuous nonlinear system (7.14) is reduced to

$$\begin{aligned}\dot{\hat{x}} &= f(\hat{x}, u, \hat{\theta}_1) + K(t)(y - \hat{y}),\\ \hat{y} &= C\hat{x},\end{aligned} \tag{7.15}$$

where $K(t) = P(t)C^T R^{-1}$ and $P(t)$ is the solution of the Riccati equation:

$$\dot{P}(t) = (A(t) + I\eta)P(t) + P(t)(A(t) + I\eta)^T + Q_R - P(t)C^T R^{-1} C P(t), \tag{7.16}$$

with the time-varying matrix $A(t) = \frac{\partial f(x,u,t)}{\partial x}$ evaluated at $\hat{x}$, Q_R and R are the noise covariance matrices. When the error

$$e = y - \hat{y}$$

converges to 0, the estimated value $\hat{L}_e$ and $\hat{\lambda}_e$ are used for the last leak evaluation.

- $\mathbf{Con - N_f}$. Leak counter. The score N_F is initialized to 0 and is incremented by 1 each time a new leak occurs. The task is achieved by searching for a positive rising edge of $(Q(0) - Q(L))$.
- $\mathbf{Cal - \hat{L}_i}$. Evaluation of the leak parameter L_i, after the filter convergence, by the static relations (7.6), (7.7) and (7.8) and considering the score N_f.

Since the static relations of $\mathbf{Cal - \hat{L}_i}$ and that the switch driven by the symptom I_r separate the signals coming from the functions $\mathbf{Idn - \hat{\theta}_1}$ and $\mathbf{Rec - \hat{L}_e}$, respectively, the estimators of each task can be independently designed. Therefore, by assuming that the algorithm starts under healthy conditions, the MSL procedure is accomplished as follows.

The identifier $\mathbf{Idn - \hat{\theta}_1}$ transfers the parameter $\hat{\theta}_1$ to the estimator $\mathbf{Rec - \hat{L}_e}$, which estimates the system state (7.14) in normal condition. This condition remains as long as the symptom I_r is off. At time t_f, when I_r switches on, the EKF estimates the equivalent leak L_e and λ_e with the fixed parameter $\hat{\theta}_1$ estimated at t_f, and the score increases by 1. Moreover, the estimated equivalent leak parameters are supplied to the function $\mathbf{Cal - L_i}$, which calculates the real parameter of the leak. Once the error S_{EKF} is less than T_{EKF}, the $\mathbf{Idn - \hat{\theta}_1}$ hold function is off, and the actual $\hat{\theta}_1$ is used by the EKF. In adition the algorithm is again ready to catch another leak.

From an operation point of view, the algorithm works in three states.

- **Detection State.** By using (7.12) with L_e and λ_e known, $\hat{\theta}_1$ is estimated, $\mathbf{Gen - I_r}$ simultaneously generates I_r and the EKF error remains close to 0. This stage remains so long as the symptom is off. When I_r turns on, $\hat{\theta}_1$ is held, and the algorithm is switched to the Location State.
- **Location State.** By using (7.15) with constant $\hat{\theta}_1$ and adding L_e and λ_e as states, the new equivalent leak is estimated. When the parameters converge, the residual I_r turns off, and the update stage starts.
- **Update State.** By using the equivalent parameters of the location stage, L_e and λ_e, the physical parameters of the last leak are calculated by (7.7), (7.8) and (7.6). Moreover, the leak score is increased by 1, and the leak parameters are stored. Thus, the algorithm returns to the detection stage, and it is ready to detect the next leak.

7.4 SCADA-II System for Sequential Leaks' Location

Let us begin from the whole SCADA-II system and finish with the main details. First of all, we have two main parts in this system (in Fig. 7.5); the pipe prototype and the SCADA (supervisory control and data acquisition) composed of the data acquisition interface and the PC with the FDI program in it.

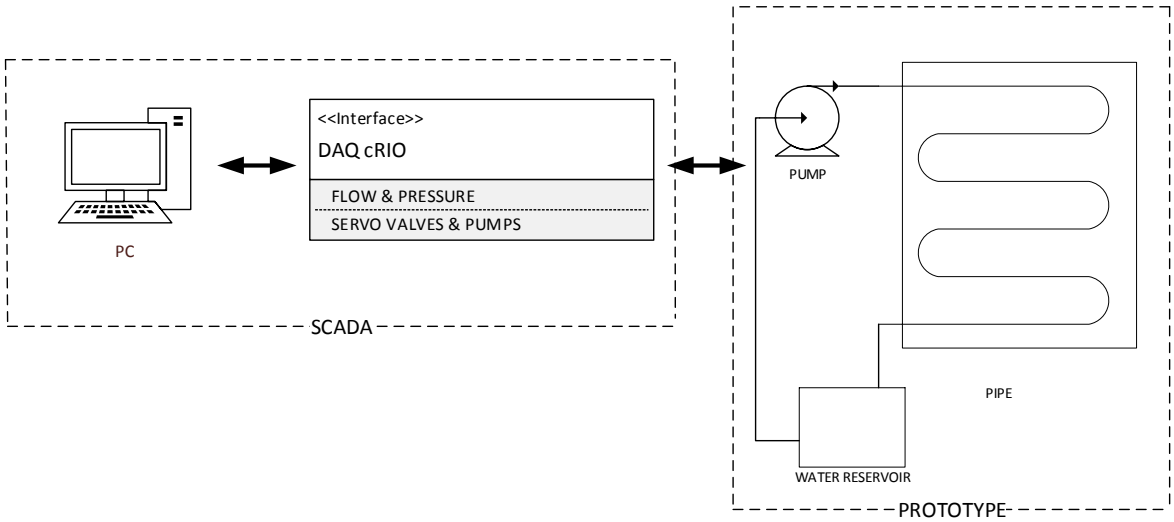


Fig. 7.5: Layout of the SCADA-II system. Arrows mean data interchange. The PROTOTYPE dashed block corresponds to the scheme of Fig. 7.2

As presented in Section 7.2, the prototype permits establishing the experiment conditions such as set point flow rate in the pipe and which valve should be opened, as well as when and how much. Those are the experiment conditions, but simultaneously there are flow rate and pressure sensors at the beginning and end of the pipe. In addition, these sensors feed the SCADA with the actual flow and pressure data, for FDI goals.

Analog data coming from the sensors need to be equal to the data processed in the computer. In order to do that, an interface is needed that converts electrical signals (4-20 mA) to binary data and vice versa. This interface is the DAQ cRIO shown in Fig. 7.5. The interface or data acquisition system (DAQ) is integrated by a CompactRIO (cRIO) controller and two I/O (input/output) modules. Additionally, by means of this interface the SCADA commands are sent to servo valves and pumps.

The SCADA-II, Fig. 7.5, is a LabVIEW program on a personal computer. This program handles signals from and to the pipe system and runs the algorithm that detects the leaks and localizes them.

7.4.1 LabVIEW programming as a tool for SCADA-II implementation

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) is a design platform that uses a visual programming language. This visual programming, called graphical language G, was developed by National Instruments (National-Instruments, 2021), and it is mainly used for data acquisition, instrument

control and industrial automation. One of its characteristics is the data flow programming: when all data is available all the subroutines (called sub VI, where VI is the acronym for virtual instrument) or functions are executed. The graphical programming consists of wires that connect nodes creating a program called virtual instrument (VI); a node can be another sub VI or a function. LabVIEW has a whole set of functions available to the programmer. A VI has three components: a front panel, block diagram and a connector pane. The front panel is the user interface and has controls and indicators (graphs or displays). The block diagram or back panel is the graphical program associated with the source code. The connector panel has the input-output signals or data that may be connected to other VIs, controls or indicators. To emulate real-time with Labview, a VI must be executed with very small time intervals.

7.4.2 Data acquisition system

Figure 7.6 shows the data flow of the proposed SCADA-II. The kernel of the data acquisition system is the cRIO system that receives signals from sensors, sends signals to the inverter and the servo valves and communicates with the HMI (human machine interface) in the host computer (PC).

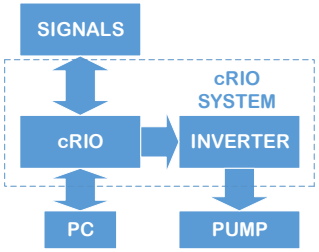


Fig. 7.6: cRIO data and signals' layout of the SCADA-II

The data acquisition system is composed of the modules NI9203 and NI9265 connected to the cRIO-9067 controller. The NI9203 module is an 8-channel current to binary data converter, and the NI9265 module is an 8-channel binary to current converter. The current loops range from 4 to 20 mA. Communication with the PC can be achieved via USB or Ethernet connector.

Figure 7.7 shows the cabinet with the inverter, the cRIO controller, the IO modules, the terminal block connector and voltage source. IO modules are connected to the controller via an internal bus and are also connected to the external devices via the terminal block connector.

7.4.3 The SCADA-II program

The main part of the SCADA-II is the software: the main program. The SCADA-II system is composed of a physical phenomenon -flow in a pipe- with sensors that retrieve the phenomenon behavior and actuators that change setpoints. All these parts, however, are nothing without a program that organizes the entire operation of the system. In the next section, we explain what the user can see and do by means of this program.

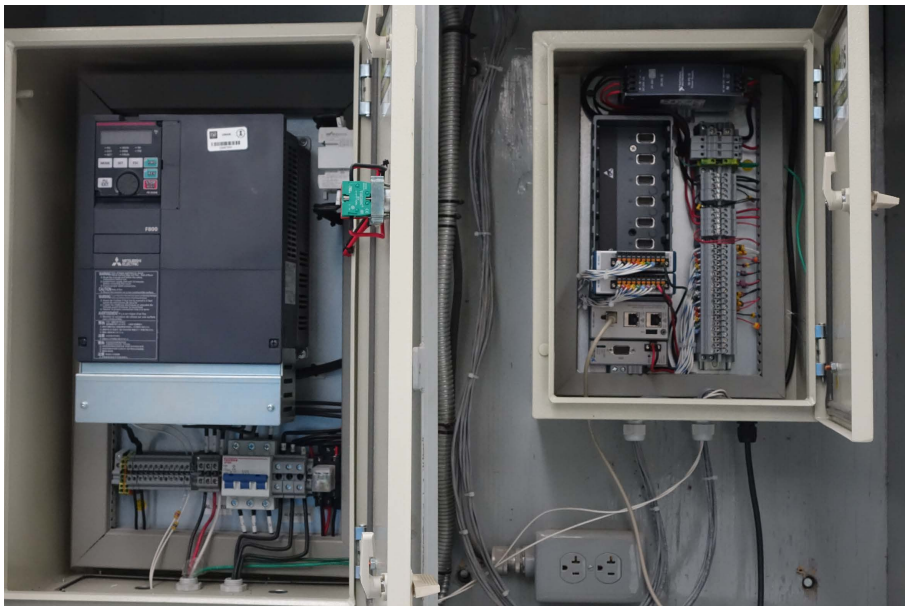


Fig. 7.7: Hardware: Inverter on the left and cRIO controller, modules and terminal block connector on the right

7.4.3.1 SCADA-II user interface

Communication between the user and the main program takes place on the user interface or HMI shown in Fig. 7.8. The screen is divided into three windows: Process **Commands**, Data **Monitoring** and **Graphs** of the estimation. The program runs with a sampling period of 100 ms.

Commands are as follows: sampling period, pump frequency, opening of the servo valve $SV1$, opening of the servo valve $SV2$ and the stop button. The sampling period in milliseconds is established prior to the system operation, by considering the time response of the system that is approximately 5 s. The aperture of servo valves can be changed any time from 0 to 100%. Pushing the stop button ends the operation, and the historic data sheet is then saved in the cRIO controller HD prior to stop main program.

The number of leaks, the time elapsed and the leak alarm are the monitored variables. Each time a new leak is detected the variable **Leak No** is incremented. The elapsed time corresponds to the time recorded from the beginning of the operation until the operation ends. The leak button is a luminous alarm that turns red when a leak appears and remains red as long as a leak exists.

In the graphs window of the screen, there are two columns. In the left column pressures, flow rates at the ends and servo valves apertures are shown. In the right column, estimated and calculated data are shown: the θ parameter associated with the hydraulic gradient, the parameters relative to the leak size and of position λ_e and L_e, respectively, and the position of the leaks z_1 and z_2. To have a steady screen, the user interface (HMI) is actualized each 5 periods, that is each 0.5 s.

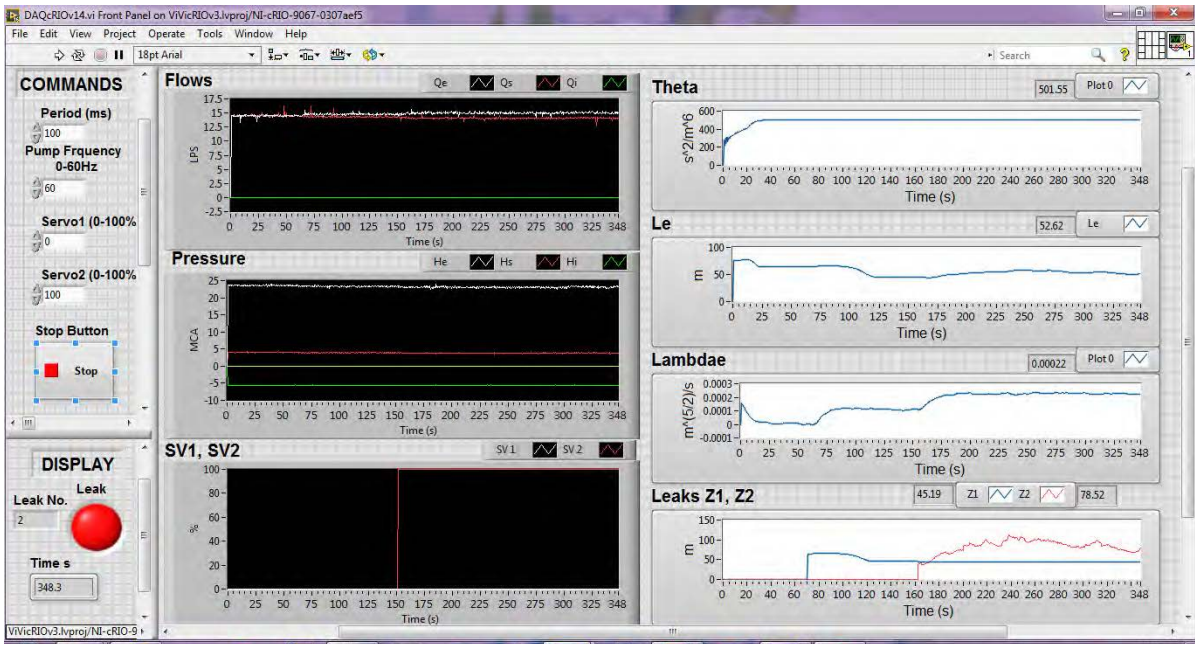


Fig. 7.8: PC screen shot of the SCADA-II HMI with leaks estimated at z_1 and z_2

7.4.4 Main program

Behind the user interface, there is a main program written in LabVIEW language. Figure 7.9 shows an overall view of the flowchart for the SCADA-II program. This describes the set of activities that take place during the process and the sequence in which they are accomplished. Furthermore, it can be divided in two main activities: 1) those that occur at the beginning of and during the operation and 2) those that correspond to parameter identification and leaks' location. In the following, the states of the diagram will be thoroughly explained.

7.4.4.1 Operation setup

The program begins with setting initial values of variables and constants, and the user establishes the sample time period. This time period is related to the time of the loop that contains the main program. The block *if* of the flowchart shown in Fig. 7.10 (a) indicates when the loop execution starts. Figure 7.10(b) shows the flowchart of the program that interacts with the user. The program can be ended any time when the user presses the stop button. Right after pressing the button, all data recorded are downloaded to a data sheet that will be saved on the hard disk in the cRIO controller. If the program continues, the hydraulic pump velocity can be changed any time in the interval from 30 to 60 Hz. The pipeline prototype has two servo valves that can be actuated from the program in the range from 0 to 100% of aperture.

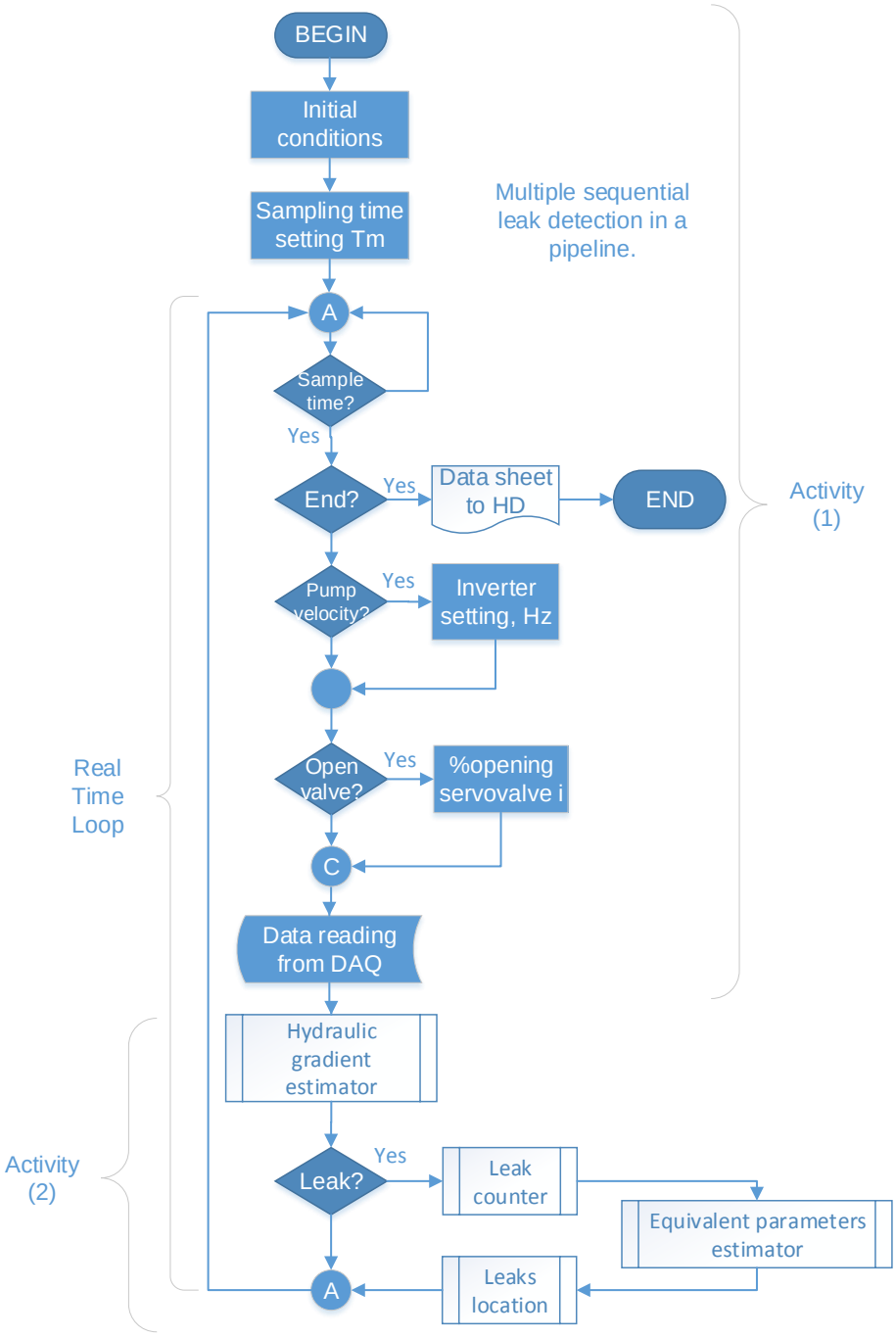
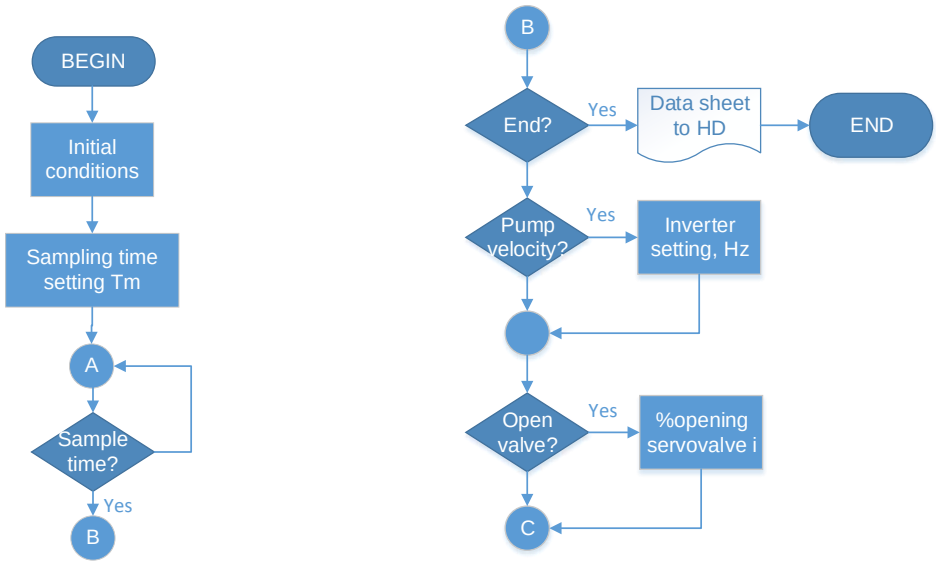


Fig. 7.9: Flowchart of the SCADA program

7.4.4.2 Parameter identification and leaks location

Figure 7.11 shows the flowchart of the main program for leak diagnosis. It begins with the data reading from the data acquisition system (DAQ). The reading process is shown in Algorithm 4. This is a subrou-



(a) Setting parameters and initial conditions (b) Operation point, pump and valves settings

Fig. 7.10: Flowcharts of subprograms: setting operating data and user interaction

tine that runs in the FPGA (field programmable gate array) area of the cRIO controller, where reading and writing data are available in fact. AO_i are the analog outputs, and AI_i are the analog inputs.

Algorithm 4 Target program in FPGA: sensors reading and commands to inverter and servo valves

t

Input: $T_s, AO_i, i = 1,\ldots,4$

Output: $A_i, i = 1,\cdots,6$

Set rank (0-20 mA) in AIs

for k **do**

$[AI_1,...,AI_6](k) \rightarrow FIFO$

$AO_0(k) \leftarrow F_{pump}$

$AO_1(k) \leftarrow Servo \quad 1$

$AO_2(k) \leftarrow Servo \quad 2$

$AO_3(k) \leftarrow Servo \quad OUT$

end for

The second block of the flowchart shown in Fig. 7.11 is the *hydraulic gradient estimator*, that is characterized by $(J(Q,\theta_1) = \theta_1 Q^2)$ described in Section 7.3. The sub VI that performs this task is shown in Fig. 7.12, and Algorithm 5 shows the parameter estimation implementation.

Returning to the flowchart shown in Fig. 7.11, the next block is the *Leak* condition; if there is no leak, the flow returns to node A and waits for a new sample period. If there is a leak, a leak counter and location loop follows. The system identifies a leak when residual $r(t)$ is greater than its threshold th.

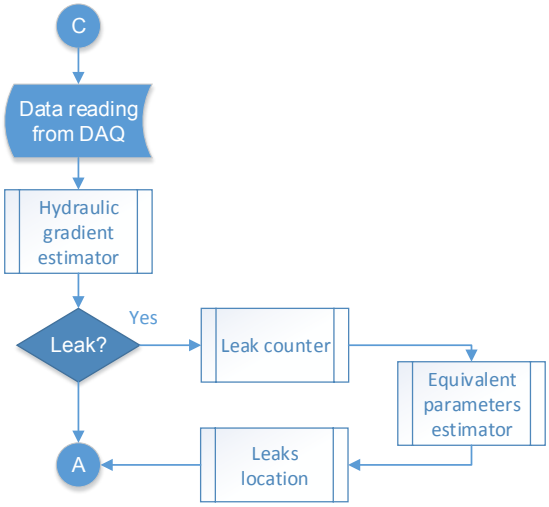


Fig. 7.11: Flowchart of parameter estimation and leaks location

Algorithm 5 Pressure gradient parameter estimation $\hat{\theta}$ (7.12)

Input: $H_0, H_L, Q_0, Q_L, I_r, L_e$
Output: $\hat{\theta}_1$

Set k_y, k_θ
if $I_r = 0$ **then**
$\hat{y} = \alpha(\hat{y},u,t) + \beta(\hat{y},u,t)\hat{\theta} - k_y(\hat{y}-y)$
$\dot{\hat{\theta}} = -k_\theta \beta(\hat{y},u,t)^T(\hat{y}-y)$
else[$I_r = 1$]
Hold $\hat{\theta}$
end if

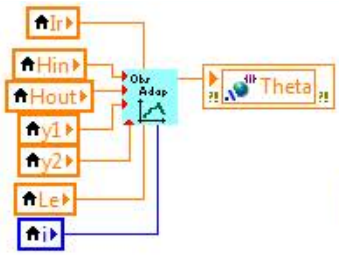


Fig. 7.12: sub VI for θ identification

To avoid false leak alarms, an adaptive threshold for $r(t)$ is implemented. This adaptive threshold is an improvement of proposal $th(t) = a_{t1} + a_{t2}s(t) + a_{t3}|\dot{s}(t)|$, given by Isermann (2011), where $s(t)$ is a reference signal taken from the system, Q_0 is used in this implementation. The improvement here consists of removing the absolute value of the derivative. The reason for the change is because if one considers the flow rate $Q_0(t)$ as the reference signal, $\dot{Q}_0(t)$ could be negative if noise, uncertainties or operation point changes exist. This condition then produces that the threshold still increases the creation

of false alarms. Thus, the novel adaptive threshold of the SCADA-II is given by

$$th(t) = a_{t1} + a_{t2}Q_0(t) + a_{t3}\dot{Q}_0(t), \tag{7.17}$$

with a_{t1}, a_{t2} and a_{t3} empirical values set by a trial and error process. As a consequence, when $r(t) > th(t)$ a binary flag alarm is activated to avoid the increment of the leak counter when no leak occurs.

The loop of estimation of Fig. 7.11 begins with the *Leak counter* block where the residual $r(t)$ is analyzed, and each time a rising edge appears, a new leak has occurred, and the counter L_c, is incremented.

The next block is the *Equivalent parameters estimator* where the EKF is implemented to find the equivalent parameters and the binary signal. The LabVIEW sub VI that performs this task is shown in Fig. 7.13. Algorithm 6 shows the sequence of operations that occur in this VI. It begins by solving a Riccati equation, then finds the equivalent parameters L_e and λ_e and sets the binary signal I_r.

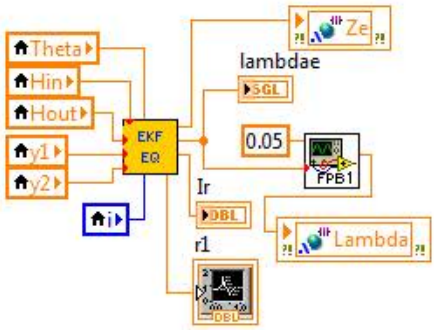


Fig. 7.13: sub VI for equivalent parameter identification by *EKF*

The last block of Fig. 7.11 which is the *Leaks location*, is a process that depends on leak counter value and estimated equivalent parameters and starts assuming the first leak $z_1 = L_1$ as $\hat{L}_e$. Figure 7.14 shows the sub VI that generates the distance L_2 of the second leak by assuming that first leak is already known and calculates the position of the second leak $z_2 = L_1 + L_2$. The commands that perform the location are shown in Algorithm 7.

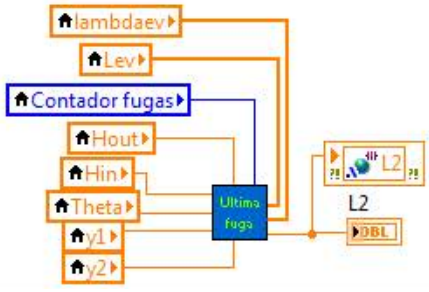


Fig. 7.14: sub VI for leak location

Algorithm 6 L_e and λ_e estimation and I_r setting

Input: $H_0, H_L, Q_0, Q_L, \theta_1, th$
Output: $\hat{L}_e, \hat{\lambda}_e, I_r$
Set Q_R, R, T_{EKF},
$r = Q_0 - Q_L$, $e_1 = Q_0 - \hat{Q}_0$, $e_2 = Q_L - \hat{Q}_L$, $S_{EKF} = |e_1| + |e_2|$
if $r > th$ **then**
  if $S_{EKF} > T_{EKF}$ **then**
    Solves Riccati equation for $\dot{P}$, Eq. (7.16)
    $K(t) = P(t)C^T R^{-1}$
    Solves $\dot{\hat{x}}, \hat{y}$, Eq. (7.15)
    Solves $\hat{\lambda}$, Eq. (7.13)
  else
    Hold $\hat{\lambda}, \hat{L}_e$
  end if
else
  Hold $\hat{\lambda}, \hat{L}_e$
end if
if $r > th$ **then**
  if $S_{EKF} > T_{EKF}$ **then**
    $I_r = 1$
  else
    $I_r = 0$
  end if
else
  $I_r = 0$
end if

Algorithm 7 Leaks location $z_1 = L_1$ and $z_2 = L_1 + L_2$

Input: $\hat{L}_e,\ \hat{\lambda},\ L_c,\ Q_0, H_L,\ Q_0,\ H_L$
Output: $z_1,\ z_2$
if $L_c = 0$ **then**
  $z_1 = 0, z_2 = 0$
else
  if $L_c = 1$ **then**
    $z_1 = \hat{L}_e$
  else
    if $L_c = 2$ **then**
      Solves Eq. (7.6) for L_2
      $z_2 = z_1 + L_2$
    end if
  end if
end if

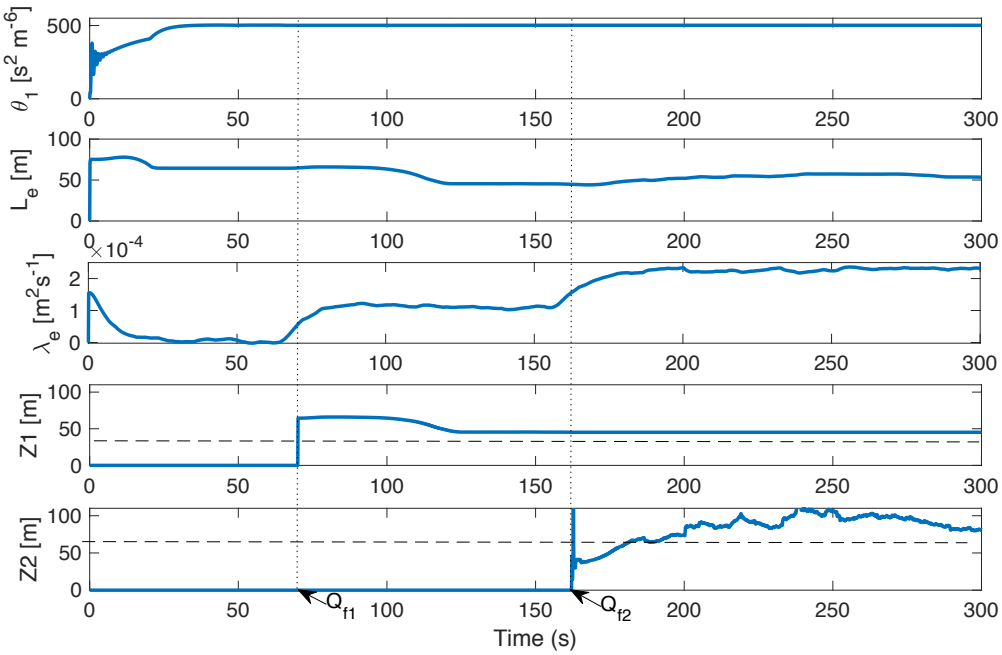


Fig. 7.15: Estimated parameters and leaks location. Arrows indicate the occurrence time of the leaks. Dashed lines are the real position of leaks

7.5 Conclusions

A remarkable finding is the success of the method for detection and localization of two consecutive leaks implemented in this SCADA-II, which operates in real-time. Even a Riccati equation is solved at each step.

Figure 7.15 shows the time evolution of the estimation plotted with the help of MATLAB from on-line data. The experiment consists of two sequential leaks caused with the valves V2 and V3. Upper graphs show the estimated parameters associated with the equivalent leak, and the lower two graphs show the physical position of the leaks.

Some characteristics are to remark of the hydraulic system. First of all is, even though the system is working in steady state: signals are not completely steady, they are changing around a mean value. This is because pumps, discontinuities inside the pipeline and continuous recycling of the leakage flow causes disturbances in the system.

Another remark is that the parameter estimation requires around 30 s to reach the final value. As a consequence, this time interval constrains how close in time the leaks can occur.

With respect to the performance of the novel adaptive threshold, diverse experiments have shown that the alarm signal is activated with an outflow of the leaks less than 1% with respect to the nominal flow rate. Figure 7.16 shows the front panel of the corresponding sub VI. One can see the behavior of the residual versus the adaptive threshold when a leak and a change in the operating point (OP) are present. The leak appears at 58 s and the change in OP at 115 s. The adaptability of the threshold, as a function of $Q(0)$, can be seen when the operating point is changed by the pump frequency from 60 Hz to 55 Hz, for a while residual goes down so the threshold.

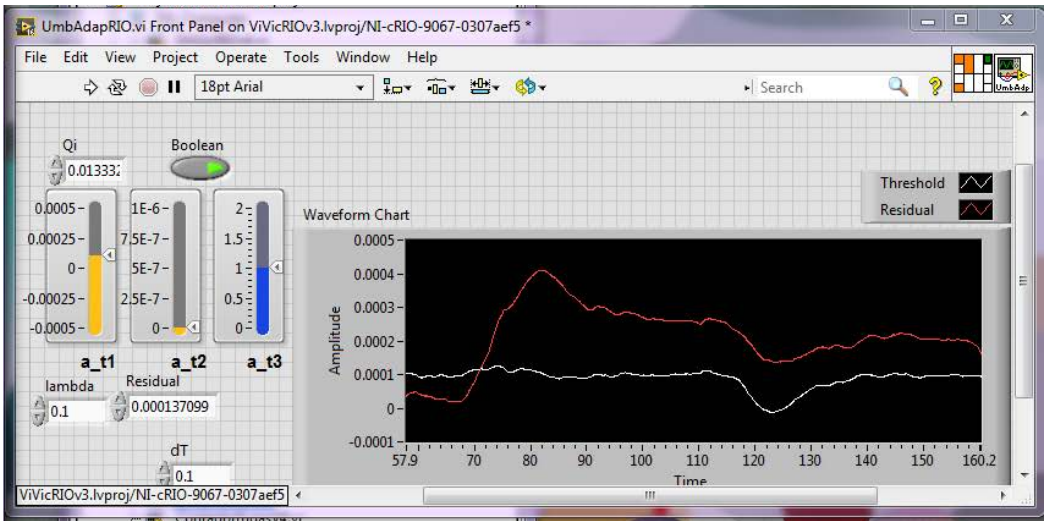


Fig. 7.16: Adaptive threshold front panel

As a conclusion, the goal of designing a SCADA at UNAM and its validation for sequential leaks detection and localization was fully achieved. Nevertheless, work remains to be done for future versions. The settling time of the estimators and location error must be reduced.

References

Aamo, O. M and Salvesen, J and Foss, B. (2006). Observer Design Using Boundary Injection for Pipeline Monitoting and Leak Detection. In *ADCHEM 2006, IFAC*, pages 53–58.

Bansal, R. (2005). *Fluid mechanics and hydraulic machines*. Laxmi Publications (P) LTD.

Barmesa (2018). *Centrífugas mediana presión*. Barmes pumps.

Besançon, G. (2007). *Nonlinear Observers and Applications*, chapter Parameter/Fault Estimation in Nonlinear Systems and Adaptive Observers, pages 211–221. Springer.

Billman, L. and Isermann, R. (1987). Leak detection methods for pipelines. *Automatica*, 23(3):381–385.

Carrera, R. (2019). *PROTOTIPO PARA DETECCIÓN DE FALLAS EN TUBERÍAS: MANUAL DE USO*. Instituto de Ingeniería, UNAM.

Carrera, R. and Verde, C. (2001). Localizador Automático de Fugas en un Ducto. *Ingeniería Hidraulica en México*, XVI:139–151.

Chaudhry, M. C. (2014). *Applied hydraulic transients*. Springer: New York.

E+H (2018a). *Cerabar s pmp71*. Endress + Hauser.

E+H (2018b). *Proline promass 80F, 83F*. Endress + Hauser.

Fang, W., Adegboye, M., and Karnik, A. (2019). Recent advances in pipeline monitoring and oil leakage detection technologies: Principles and approaches. *sensors*, 19(2548).

GF (2018). *Type 130/133 electrically actuated ball valve*. Georg Fischer Piping Systems.

Isermann, R. (2006). *Fault-Diagnosis Systems*. Springer, Berlin.

Isermann, R. (2011). *Fault Diagnosis Applications*. Springer Verlag.

Liu, C., Li, Y., and Xu, M. (2019). An integrated detection and location model for leakages in liquid pipelines. *Journal of Petroleum Science and Engineering*, 175:852–867.

Mitsubishi (2018). *Inversores freqrol*. Mitsubishi Electric.

Moubayed, A., Sharif, M., Luccini, M., Primak, S., and Shami, A. (2021). Water Leak Detection Survey: Challenges & Research Opportunities Using Data Fusion & Federated Learning. *IEEE Access*.

Narendra, K. S. and Annaswamy, A. M. (1987). Persistent excitation in adaptive systems. *International Journal of Control*, 45(1):127–160.

National Instruments (2021). ¿qué es labview? https://www.ni.com/es-mx/shop/labview.html.

NI (2018). *Qué es compactRIO*. National Instruments.

Panasonic (2013). *CF-53 Reference Manual*. Panasonic Corporation.

RC (2018). *Remote control electric actuators*. Remote control.

Reif, K., Sonnenmannn, F., and Unbehauen, R. (1998). An EKF-Based Nonlinear Observer with a Prescribed Degree of Stability. *Automatica*, 34(9)(9):1119–1998.

Rojas, J. (March, 2021). *DIAGNÓSTICO DE FUGAS SECUENCIALES*. PhD thesis, Posgrado en Ingeniería UNAM.

Rojas, J., Verde, C., and Torres, L. (June 2021). Estimation of hydraulic gradient for a transport pipeline. *ASME. J. Pressure Vessel Technol*, 143(3):031801.

Sieber, H. and Isermann, R. (1977). Leckerkennung und lokalisierung bei pipelines durch on-line-korrelation mit einem prozessrechner. *Regelungstechnik*, 25:69–74.

Verde, C., Bornard, G., and Gentil, S. (2003). Isolability of Multi-leaks in a Pipeline. In *Proceedings 4th MATHMOD, Vienna*, Vienna, Austria. ARGESIM Verlag, ISBN-3-901608-24-9.

Verde, C. and Rojas, J. (2017). *Modeling and Monitoring of Pipelines and Networks*, chapter Recursive Scheme for Sequential Leaks' Identifications, page 134. Springer.

Verde, C. and Torres, L. (2017). *Modeling and Monitoring of Pipelines and Networks*. Springer.

Zhang, T., Tan, Y., Zhang, X., and Zhao, J. (2015). A novel hybrid technique for leak detection and location in straight pipelines. *Journal of Loss Prevention in the Process Industries*, 35:157–168.

Chapter 8
Observer-based calibration of a power-law model for head loss in pipelines

Lizeth Torres[1] and Cristina Verde[1]

Abstract This chapter presents a real-time approach based on a nonlinear state observer for estimating both the coefficient and the exponent of a power-law model for head loss in a pipeline. Such a power law is nothing but a generalization of conventional models such as the Darcy-Weisbach and the Hazen-William formulas. The interest in estimating the parameters of the power law is twofold: to avoid handling values of physical parameters that may be uncertain, such as relative roughness, and to have an appropriate model for computational pipeline monitoring applications, such as leak detection and localization. The real-time estimation is performed by a state observer from the following available information of the pipeline: a flow rate and two pressure head recordings. To demonstrate how the proposed approach works, the state observer was implemented in MATLAB and experimental data from a lab pipeline was used to feed it. In addition, numerical simulations of a leak location algorithm are presented to show how a well-calibrated head loss model is highly necessary for leak diagnosis.

8.1 Introduction

The proper management of pipelines and fluid distribution networks requires some tasks that should be preferably executed in real time, for example, the detection and identification of faults or the pressure control for minimizing fluid losses (Verde and Torres, 2017; Puig et al., 2017). Usually these tasks are performed by algorithms based on mathematical models that are formulated from physical principles. In addition, although these algorithms have been shown to work very well in practice, they have an intrinsic problem: they must be frequently updated because their governing equations are functions of physical parameters that evolve over time in a slow or drastic way according to the operational and environmental conditions. In particular, in pipelines that have been in service for a considerable time, there are two physical parameters that notably change: the roughness and the internal diameter. Both in fact can be related to a normalized parameter: the relative roughness.

Relative roughness varies because of the natural deterioration processes that affect the internal wall of the pipelines such as corrosion, erosion, tuberculation or the deposit of materials. Such a variation directly modifies the dissipation of energy between the wall and the fluid.

[1]Instituto de Ingeniería UNAM, Ciudad Universitaria, Coyoacán, 04510, Mexico, e-mail: ftorreso@iingen.unam.mx and verde@unam.mx

The total energy dissipation in a pipeline is the sum of two hydraulic losses: the quasi-steady and the unsteady losses (Adamkowski and Lewandowski, 2006). Several authors have tried to model the unsteady loss, which is present in fast transients, by proposing in parallel, models that match experimental data and methods to calibrate them. These models and the associated calibration methods, however, are not discussed here since the unsteady loss is out of the scope of this chapter. The reader may consult the reviews provided in Bergant et al. (2001) and Savic et al. (2009) to know more about the state-of-the-art research on this topic.

The quasi-steady loss, which is known as head loss or pressure drop, can be divided into two main classes: major losses, associated with energy loss caused by viscous friction in the pipeline wall, and minor losses, associated with bends, fittings and valves. The head loss can be then expressed as follows (White, 1999):

$$\Delta H(Q) = \Delta H_f(Q) + \sum_{i=1}^{N} \Delta H_m(Q), \tag{8.1}$$

where Q is the flow rate in the pipeline, $\Delta H_f(Q)$ denotes the major losses and $\sum_{i=1}^{N} \Delta H_m(Q)$ represents the sum of the minor losses.

The conventional equations used to calculate the major losses are the Darcy-Weisbach (DW) equation together with more empirical formulas such as the Hazen-Williams (HW) and the Manning equations, which are power laws (Moore, 1959). The DW equation, however, is more accepted with respect to the empirical formulas because it is dimensionally homogeneous and applicable to water as well as to other fluids (Liou, 1998). The DW equation is expressed as follows:

$$\Delta H_f(Q) = f(Q)\frac{L}{\phi}\frac{Q^2}{2gA_r^2}, \tag{8.2}$$

where $f(Q)$ is the dimensionless friction factor, L is the length of the pipeline, ϕ is the pipeline diameter, A_r is the cross-sectional area and g is the gravitational acceleration. Minor losses represent additional energy dissipation provoked by pipeline devices that act against the fluid and reduce its energy, velocity, or momentum.

The major and minor losses can also be simultaneously calculated with the DW equation, but the real length L involved in the DW equation must be replaced by a equivalent length that depends on the flow rate and characteristics of the pipeline accessories (Chapter 2, Larock et al. (1999)).

As it can be seen, the DW equation is a function of the flow rate and other physical parameters such as the friction factor, which is also a function of the flow rate (through the Reynolds number) and additional parameters, such as the relative roughness and the kinematic viscosity. For laminar flow, the friction factor can be calculated by using the Hagen-Poiseuille law, which is very simple (Hitzer, 2001). Nevertheless, in real operating conditions, pipelines work in the turbulent flow, and the friction computation becomes more complicated. In such a case, the consensual way to calculate the friction factor is by using the Colebrook-White (CW) equation, which is an implicit formula for which many explicit approximations have been proposed (Brkić, 2011). A mutual problem of using the CW equation or its approximations, however, is that they require the current value of the relative roughness along the pipeline. The relative roughness can be measured in real time by using a profilometer (as seen in Farshad et al. (2001); Sletfjerding and Gudmundsson (2003)), but it is necessary to wait for the complete scan of

the pipeline's internal wall. This test can be quite lengthy if the pipe is long or difficult to access, which leads to considering better alternatives for avoiding the use of the actual value of the relative roughness.

An alternative is the calibration of a model for the head loss[1]. To perform this alternative, it is necessary to provide a calibration method (a parameter estimation method) and a model of the head loss (e.g., the DW equation or the HW formula) with parameters to be experimentally estimated.

The calibration problem remains an open problem. Currently, many researchers dedicate much effort in proposing more efficient and easy-to-tune methods to convince practitioners to use them. Recently in Kumar et al. (2010), the authors presented an approach to estimate the roughness coefficients of the pipelines of a network, which is based on a k-means clustering algorithm and on the theory of graphs. In the same year, the Battle of the Water Calibration Networks (BWCN) was conducted by 14 teams from academia, water utilities, and private consultants. The BWCN results were given at the 12th Annual Conference on Water Distribution Systems Analysis and presented in Ostfeld et al. (2011). More recently, in Gao (2017) a method based on weighted least squares (WLS) was presented to compute the Hazen-Williams C factors of pipelines. In Díaz et al. (2017), the authors proposed a method based on a multi-period state estimation and mathematical programming decomposition techniques. Additionally, they presented an analysis of the observability of the roughness from available measurements.

In this spirit, this contribution proposes a real-time approach based on a state observer for calibrating the coefficient and the exponent of a power law that models the head loss only as a function of the flow rate and not as a function of the relative roughness and other uncertain parameters. The proposed approach only requires a flow rate and pressure measurements. Therefore, this method can be used in trunk pipelines instrumented with pressure and flow rate sensors placed at locations such as pump stations, where these sensors are usually installed.

The benefits of proposing a power law to approximate the total head loss together with a method for calibrating it in real time are the following: (a) to have a head loss model that expresses a general representation of conventional formulas used to compute the head loss such as the HW, the Manning or the Valiantzas equations (Chapter 2, Larock et al. (1999), Chapter 2, Chaudhry (1979)); (b) to avoid using a complicated formula such as the CW equation, which is implicit because it requires iterative methods to be solved; (c) to have a valid model for a given operation region instead of formulas for describing the complete turbulent regimen; (d) to have an identifiable model, i.e., a model with a structure involving easily estimated parameters by means of real-time algorithms; (e) to have a differentiable function.

An essential requirement in applying our approach is that the flow in the pipeline must be excited to ensure the identifiability of the parameters. This means that the flow in the pipeline must be perturbed during a short period while the parameters are estimated. Therefore, in order to satisfy the excitation condition, a steady-oscillatory flow in the pipeline is induced by injecting a sinusoidal signal into the frequency drive of the supply pump. Many of the off-line methods for detecting leaks and identifying parameters proposed by the hydraulic community, which have been verified in a laboratory, are based on the induction of steady-oscillatory flow in a pipeline by the sinusoidal maneuver of a valve (Xu and Karney, 2017).

To estimate the parameters of power laws in the context addressed here, some approaches have been proposed: for instance, those presented in Datta and Sridharan (1994) and Reddy et al. (1996), which use least squares and information (pressure and flow rate) recorded during a prescribed period. Moreover,

[1] The parameter estimation process in a model is called the calibration process in hydraulics.

the reader can find in Savic et al. (2009) a masterful review of proposed methods to calibrate pipelines and networks. The authors of this review categorize the methods as steady-state and transient and discuss the advantages and drawbacks of each one.

The novelty of this contribution is an experimental validation and the improvement of the approach presented in Torres and Verde (2018), which in turn was inspired by early results from Rojas et al. (2018), where the real-time estimation of the coefficients of both a quadratic and a cubic equation that approximate the head loss in a pipeline is proposed.

8.2 Physical system and modeling

This section describes the characteristics of the pipelines for which the proposed method can be used. This section also presents the extended model (an auxiliary model) used for the design of the state observer that performs the parameter estimation. Such an auxiliary system is deduced from a mathematical model commonly used in hydraulic problems: the rigid water column (RWC) model (Cabrera et al., 1995; Nault and Karney, 2016), which in turn is deduced from physical principles.

8.2.1 Physical system

The approach presented in this chapter works for pipelines that satisfy the following: unidirectional flow and constant cross-sectional area. Furthermore, such pipelines must be instrumented such that pressure head measurements at the ends of the pipeline are available, as well as a flow rate recording at any point of the pipeline.

8.2.2 Rigid water column model

In this chapter, the model considered for representing the head loss in a pipeline is the power law $\Delta H = \Omega Q^{\gamma}$, where Ω and $\gamma \leq 2$ are parameters related to the friction losses and fluid properties.

Since the goal of this chapter is to propose a real-time method for experimentally estimating Ω and γ by using a state observer, a dynamical model involving these parameters is needed. This model is the so-called RWC model, which describes the flow in one dimension by ignoring the compressibility of the fluid and the elasticity of the conduit such that the entire column of fluid is assumed to move as a rigid body. The RWC model can be used to describe short pipelines or small sections of long-distance pipelines. For describing the overall behavior of the flow in pressurized long-distance pipelines, the fluid should be considered as compressible or slightly compressible. By assuming slight compressibility, the model must be modified to consider the unsteady friction, which involves the pressure wave propagation velocity and pressure changes along the spatial domain. This is out of the scope of this chapter.

The RWC model for a pipeline without extractions is expressed by the following equation (Islam and Chaudhry, 1998; Nault and Karney, 2016):

$$\dot{Q} = \beta \Delta H - \alpha Q^{\gamma+1}, \tag{8.3}$$
$$y_m = Q,$$

where y_m is the measured flow rate, β and α are parameters with appropriate units of measure.

If $\dot{Q} = 0$, Eq. (8.3) becomes an expression for the head loss

$$\Delta H = \frac{\alpha}{\beta} Q^{\gamma+1}, \tag{8.4}$$

which, by defining $\Omega = \alpha/\beta$, becomes

$$\Delta H = \Omega Q^{\gamma+1}, \tag{8.5}$$

which is nothing but the head loss model to be updated experimentally in real-time. Therefore, to obtain an estimation of Ω, α and β must first be estimated.

Once Ω and γ are estimated, these can be associated with the physical characteristics of both the pipeline and fluid by means of conventional formulas for head loss, e.g., the Darcy-Weisbach equation, the Hazen-Willian equation or the Gauckler–Manning–Strickler formula. If the Darcy-Weisbach is used for a pipeline with length L, then $\gamma = 1$ and $\Omega = f(Q)L/2g\phi A_r^2$.

8.3 Real-time calibration approach

In order to estimate α, β and γ by using a state observer, these parameters can be considered as state variables and they are added to the state vector of the RWC model given by (8.3). The above results in an extended model that combines the equations governing the evolution of the original state and the equations describing the evolution of the parameters.

8.3.1 Extended model

The extended model is obtained by defining the following states: $x_1 = -\ln(Q)$, $x_2 = \beta$, $x_3 = \alpha Q^{\gamma}$ and $x_4 = \gamma$. In addition, these states form the state vector $x = [x_1\ x_2\ x_3\ x_4]^T$. The resulting extended model reads as follows:

$$\dot{x} = \begin{pmatrix} 0 & -u_1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ x_3 x_4 u_2 \\ 0 \end{pmatrix},$$
$$y = \begin{pmatrix} 1\,0\,0\,0 \end{pmatrix} x, \tag{8.6}$$

where $u_1 = \Delta H/Q$ and $u_2 = \dot{Q}/Q$ are assumed to be the inputs that form the input vector $u = [u_1\ u_2]$ and the output variable is x_1.

Remark 1: The system given by Eq. (8.6) has the following general structure:

$$\begin{aligned} \dot{x} &= A(u)x + B(u,x), \\ y &= Cx, \end{aligned} \tag{8.7}$$

where $A(u) \in \mathbb{R}^{n\times n}$ depends only on inputs and $B(u,x) \in \mathbb{R}^{n\times 1}$ is a nonlinear vector with a triangular structure (Besançon, 2007).

Therefore, one proposes in the following paragraphs a state observer for a system with the structure given by Eq. (8.7).

8.3.2 State observer

For the state estimation of extended dynamical systems with the form given by Eq. (8.7), it was proven in Torres et al. (2012) that with a specific excitation and under the usual technical (Lipschitz) assumption for a high-gain design, one can obtain an asymptotic estimation of the extended state with the following *high-gain* observer:

$$\begin{aligned} \dot{\hat{x}} &= A(u)\hat{x} + B(u,\hat{x}) + SC^T(y-\hat{y}), \\ \hat{y} &= C\hat{x}, \end{aligned} \tag{8.8}$$

where $\hat{x}$ is the estimated extended state vector, u is the input vector, y is the measured output, and S is the solution of the Lyapunov equation

$$\dot{S} = \lambda S + [A(u) + dB_\lambda(u,\hat{x})]S + S[A(u) + dB_\lambda(u,\hat{x})]^T - SC^TCS, \tag{8.9}$$

where λ is a parameter for adjusting the convergence rate and $dB_\lambda(u,\hat{x}) = \partial B(u,\hat{x})/\partial\hat{x}$ is the Jacobian of $B(u,\hat{x})$. Particularly, for system (8.6), the Jacobian reads as follows:

$$dB_\lambda(u,\hat{x}) = \frac{\partial B(u,\hat{x})}{\partial x} = \begin{pmatrix} 0\,0 & 0 & 0 \\ 0\,0 & 0 & 0 \\ 0\,0 & u_2\hat{x}_4 & u_2\hat{x}_3 \\ 0\,0 & 0 & 0 \end{pmatrix}. \tag{8.10}$$

Remark 2: Notice that $\hat{\alpha}$ can be obtained as $\alpha = \hat{x}_3/(-\exp(\hat{x}_1))^{\hat{x}_4}$.

Remark 3: Since the estimations of $\hat{\alpha}$, $\hat{\beta}$ and $\hat{\gamma}$ from the state observer are quantities that evolve over time and may contain both noise and unmodeled effects, the mean of these estimations must then be calculated once these estimations reach steady state in order to have a unique value: $\bar{\alpha}$, $\bar{\beta}$ and $\bar{\gamma}$. The variation of the estimates from the mean values can be calculated as follows: $e_\alpha = \bar{\alpha} - \hat{\alpha}$, $e_\beta = \bar{\beta} - \hat{\beta}$ and $e_\gamma = \bar{\gamma} - \hat{\gamma}$ to analyze the unmodeled effects.

A condition for the convergence of the estimation is that inputs u_1 and u_2 must be persistently exciting: they need to satisfy a persistent or excitation condition, which is complicated to fulfill in advance. Therefore, inputs are usually designed heuristically and checked afterwards. Recently, some optimization-based methods for characterizing the persistent inputs for a particular class of systems have been proposed; see Scola et al. (2018) for instance. These methods, however, need to be extended to other classes of nonlinear systems.

8.3.3 Persistent condition: Steady-oscillatory flow

In this contribution, to obtain inputs that satisfy the persistent condition, the procedure detailed in Besançon (2016) was used, which states that a persistent input may be chosen on the basis of the output derivatives' mapping. Therefore, the mapping for system (8.6) was calculated, which reads as follows:

$$\Phi(x(t),u(t),y(t)) = \begin{pmatrix} y(t) \\ \dot{y}(t) \\ \ddot{y}(t) \\ y^{(3)}(t) \end{pmatrix} = \begin{pmatrix} x_1 \\ -x_2u_1 + x_3 \\ -x_2\dot{u}_1 + x_3x_4u_2 \\ -x_2\ddot{u}_1 + x_3x_4^2u_2^2 + x_3x_4\dot{u}_2 \end{pmatrix}. \tag{8.11}$$

The mapping (8.11) becomes injective as soon as u_1, u_2 and their derivatives are nonzero. Therefore, for any input with period T, the excitation condition is validated in a time window given by the period T. Sinus-like pressure heads and flow rate are natural candidates for satisfying such an injectivity. Moreover, many of the off-line methods for detecting leaks and identifying parameters proposed by the hydraulic community, which have been verified in a laboratory, are based on the induction of steady-oscillatory flow in a pipeline by the sinusoidal maneuver of a valve (Xu and Karney, 2017).

8.3.4 Derivative of the flow rate measurement

For implementing the state observer (8.8), the derivative of the flow rate Q measurement is required. To obtain such a derivative, the following two steps are proposed, taking into account both the periodic nature of the signal and the knowledge of its fundamental frequency.

1. Estimate on-line the Fourier coefficients that approximate the signal. This task can be achieved by using the following proposed observer, which is based on an auxiliary system.
   The first state of the auxiliary system (used for the observer design) is the integral of Q expressed as the finite version of the Fourier series:

$$\nu_1(t) = \int_0^t \left[\frac{a_0}{2} + \sum_{i=1}^{N} \left(a_i \cos(i\omega\tau) + b_i \sin(i\omega\tau) \right) \right] d\tau, \tag{8.12}$$

   where N denotes the total number of the frequency components taken into account to approximate the signal Q.
   By defining the Fourier coefficients a_0, a_i and b_i as the rest of the states, the auxiliary system is defined as follows:

$$\dot{\nu} = \underbrace{\begin{pmatrix} 0\ 1 & \cos(\omega t) & \sin(\omega t) & \ldots & \cos(N\omega t) & \sin(N\omega t) \\ 0\ 0 & -\omega\sin(\omega t) & \omega\cos(\omega t) & \ldots & -N\omega\sin(N\omega t) & N\omega\cos(N\omega t) \\ \vdots & & & & & \vdots \\ 0\ 0 & -\omega^N\sin(\omega t) & \omega^N\cos(\omega t) & \ldots & -N^N\omega^N\sin(N\omega t) & N^N\omega^N\cos(N\omega t) \end{pmatrix}}_{A_\omega} \nu, \quad (8.13)$$

$$y_\omega = C_\omega \nu,$$

where the state vector $\nu \in \mathrm{R}^M$ and the output vector $C_\omega \in \mathrm{R}^M$ are, respectively, defined as

$$\nu = \left[\int_0^t Q \ \ a_0 \ \ a_1 \ \ b_1 \ \ \ldots \ \ a_N \ \ b_N\right]^T,$$

$$C_\omega = [1 \ \ 0 \ \ \ldots \ \ 0].$$

Furthermore, $M = 2N+2$ is the total number of states.

The states of system (8.13), i.e., the Fourier coefficients, can then be estimated by using the following state observer:

$$\dot{\hat{\nu}} = A_\omega \hat{\nu} + K_\omega e_\omega, \quad (8.14)$$

$$\hat{y}_\omega = C_\omega \hat{\nu}, \quad (8.15)$$

where $e_\omega = \int_0^t Q - \hat{y}_\omega$, $K_\omega = S_\omega C_\omega^T$ and S_ω is calculated with the Lyapunov equation

$$\dot{S}_\omega = \mu S_\omega + A_\omega S_\omega + S_\omega A_\omega^T - S_\omega (C_\omega^T C_\omega) S_\omega, \quad (8.16)$$

where μ is the parameter for manipulating the convergence rate.

Observer (8.14) was tuned: $\mu = 1$, $S_\omega(0) = I$, where I is the identity matrix, the state initial conditions were chosen to be 0 and $N = 3$. This latter parameter was chosen in this way to consider only the more meaningful frequency components: the fundamental frequency and two harmonics.

2. Construct the derivative of Q by using the estimated Fourier coefficients as follows:

$$\hat{\dot{Q}} = \sum_{i=1}^{N} \left(-\hat{a}_i i\omega \sin(i\omega t) + \hat{b}_i i\omega \cos(i\omega t)\right), \quad (8.17)$$

where $N = 3$.

Thus, by using the two estimators (8.8) and (8.14), the implementation to obtain the parameters of the power law is summarized in the scheme given in Fig. 8.1.

8.4 Experiments

The results described below were obtained during experimental tests developed in a pipeline located at II-UNAM. A layout of the pipeline and its physical parameters are presented in Fig. 8.2 and Table 8.1, respectively. The installation consists of a storage tank of 10 [m^3] and a spiral loop on the vertical plane that returns the water to the tank. In addition, a servo valve is connected at the end of the line to

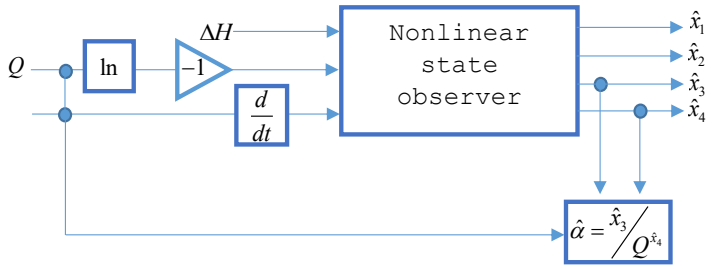


Fig. 8.1: Proposed approach

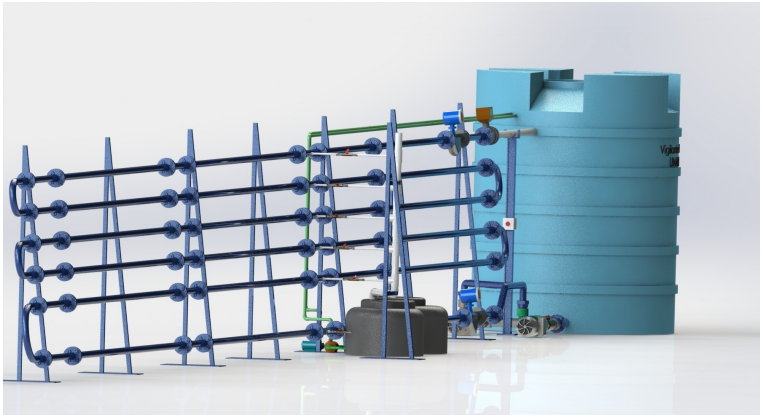


Fig. 8.2: Pipeline located at the hydrodynamics laboratory of II-UNAM

Table 8.1: Parameters of the laboratory pipeline

Parameters	Values
D [m]	0.076
L [m]	163.715
g [m/s^2]	9.81
b [m/s]	1330
ν [m^2/s]	10^{-6}@20°C

restrict the water flow. Two hydraulic pumps of variable speed fix the upstream boundary condition of the system: Pump 1 with a maximum power of 7.5 [HP] and Pump 2 with a maximum of 10 [HP]. Only Pump 1 was used here by changing the operation points with a variable frequency driver. Six valves are installed to emulate leaks or branches. The pipeline is instrumented with two flowmeters and two pressure sensors at the extremes. The Promass 83F flowmeters are based on the Coriolis principle with a maximum error of ±0.05%, while the Cerabar PMP71 pressure sensors are based on the deflection measurement of a diaphragm with a precision of 0.05%. All data during the experiments were sampled at 100 [ms].

In what follows, the description and analysis of each of the elements involved in the implementation and execution of the proposed method are presented.

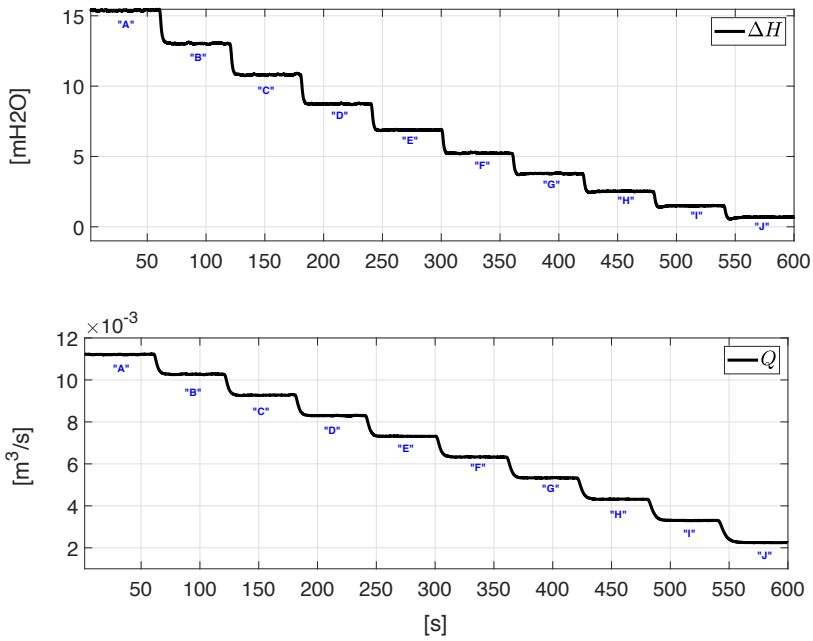


Fig. 8.3: Pressure head and flow rate recordings taken during the changes of operating point

8.4.1 Steady-state flow

Table 8.2: Steady-state operating points (OPs)

OP	$\bar{Q}$ [m^3/s]	$\overline{\Delta H}$ [mH2O]
A	0.0112	15.3839
B	0.0103	13.0320
C	0.0093	10.8155
D	0.0083	8.7336
E	0.0073	6.8807
F	0.0063	5.2348
G	0.0053	3.7866
H	0.0043	2.5240
I	0.0033	1.4993
J	0.0022	0.6865

To capture and show the behavior of the head loss as a function of the flow rate, the pipeline was set at different operating points by manipulating the supply pump through a variable frequency drive. Figure 8.3 shows the recordings taken during the experiments: the flow rate, which was measured upstream in the pipeline, and the head loss calculated from pressure records measured upstream and downstream in the pipe. The staggered form of the recorded signals reveals how the change of operating points was performed: from the highest possible point when the pump operates at the electrical network's frequency to the lowest point when the flow measurement was still reliable. The operation points are labeled with the letter A, B, C,...,I. Once the data acquisition was concluded, the mean of the flow rate measurement at each operating point, denoted hereafter as $\bar{Q}$, was obtained as well as the mean of the head loss, denoted as $\overline{\Delta H}$. Table 8.2 contains the values of $\bar{Q}$ and $\overline{\Delta H}$.

8.4.2 Steady-oscillatory flow

One way to generate steady-oscillatory flow in the laboratory pipeline is by injecting a sinusoidal signal into the pump's variable frequency drive (VFD). Taking this fact into account, two different steady-oscillatory flows were generated for testing the proposed calibration approach. The frequency of both sinusoidal signals injected into the pump's VFD was the same: $\omega = 0.1\pi$ [rad/s] (i.e. $f = 0.05$ [Hz]). The amplitude and offset of the hydraulic variables associated with both steady-oscillatory flows are listed in Table 8.3.

Figure 8.4 shows the head loss as a function of the flow rate for both the *high flow* and the *low flow*. This figure highlights the fact that oscillatory flows only cover a range of the total range of all the operating points used to show the head loss behavior. Figure 8.5 shows the derivative of the flow rate corresponding to the *high flow*.

Table 8.3: Features of the generated steady-oscillatory flows

Variable	*Low flow*	*High flow*
Q	Offset: 0.007351 [m^3/s] Amplitude: 0.0032 [m^3/s]	Offset: 0.00932 [m^3/s] Amplitude: 0.0032 [m^3/s]
H_{in}	Offset: 11.15 [mH2O] Amplitude: 8.192 [mH2O]	Offset: 16.02 [mH2O] Amplitude: 9.75 [mH2O]
H_{out}	Offset: 4.03 [mH2O] Amplitude: 1.43 [mH2O]	Offset: 5.00 [mH2O] Amplitude: 1.84 [mH2O]

8.4.3 Tuning of the state observer

Two estimations of the power-law parameters were performed. The first by using the *high flow* as the excitation and the second one by using the *low flow*. Both estimations were executed by using the same tuning parameters for the state observer (8.8). The observer was tuned by setting $\lambda = 1$. Its initial condi-

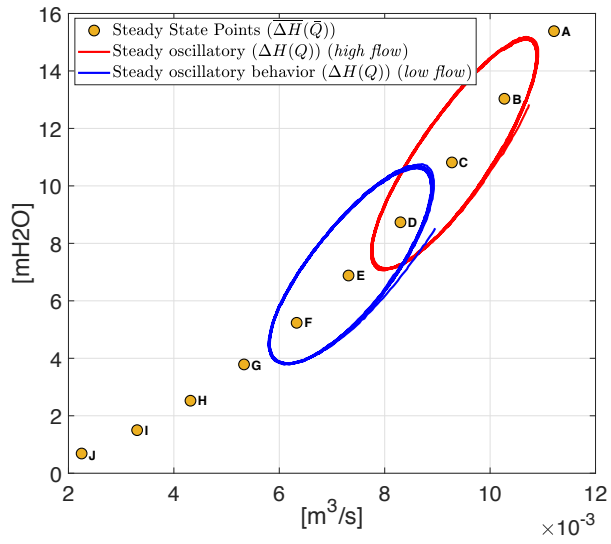


Fig. 8.4: Head loss versus flow rate in steady state and steady-oscillatory state

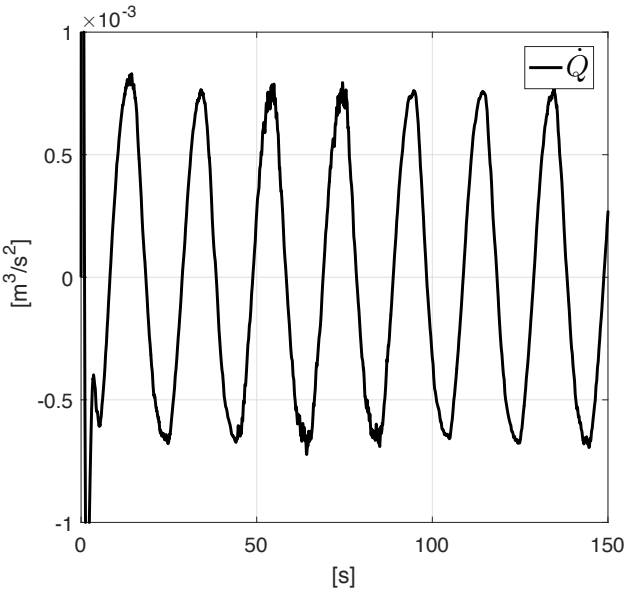


Fig. 8.5: $\dot{Q}$ in a time window of its computation for the *high flow*

tions were set as follows: $\hat{x}(0) = \begin{pmatrix} 0.12 & 0.001 & 0.01 & 0.08 \end{pmatrix}^T$ and $S(0) = I$. The observer was implemented in MATLAB/Simulink® by using the ODE3 solver with $\Delta t = 0.001$ [s] as the time step. Figure 8.6 shows the time evolution of $\hat{\beta}$ and $\hat{\gamma}$ for the *high flow* and the *low flow*. The means of the estimated parameters for the *high flow* and *low flow* were very close. In other words, the estimation results are slightly insensitive to the offset of the steady-oscillatory flow. Therefore, only the means corresponding to the *high flow* are considered in the rest of the discussion, which indeed resulted as follows: $\bar{\gamma} = 0.8897$,

$\bar{\beta} = 2.088 \times 10^{-4}$ and $\bar{\alpha} = 14.35$. A consequence of these values is $\bar{\Omega} = \bar{\alpha}/\bar{\beta} = 74569$, which is the coefficient of the estimated head loss given as $\widehat{\Delta H} = 74569 Q^{1.8897}$.

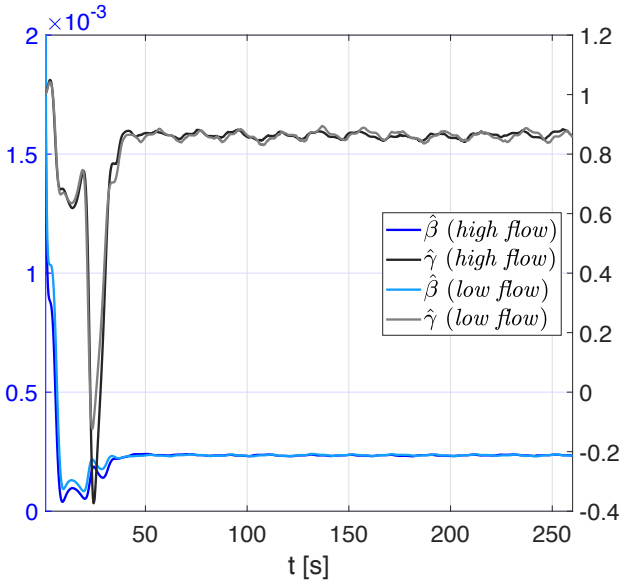


Fig. 8.6: Time evolution of $\hat{\beta}$ and $\hat{\gamma}$ by using the *high flow* and the *low flow* as excitation

8.5 Analysis of the results

In order to evaluate the performance of both the power law and the proposed algorithm, two evaluations were performed.

8.5.1 Model comparison

The first is a comparison of the errors calculated from real data and several models of the head loss in steady state. These models are as follows: the Hazen-Williams (HW) formula, a quadratic equation without the linear term and a quadratic equation. The coefficients of the head loss models were also estimated by using state observers. The errors are calculated as the difference between the experimental data and the estimated with each formula, i.e., $e = \Delta H - \widehat{\Delta H}$. The resulting errors are presented in Table 8.4. The smallest errors are written in bold. Notice that the power law approximates with better accuracy the behavior of the head loss at almost all the operation points. To corroborate this fact, a visual comparison can be performed in Fig. 8.7, which concretely shows the estimated head loss (with each formula) normalized with respect to the real head loss. It is noticeable that the power law and the HW formula approximate with better accuracy the head loss behavior.

Table 8.4: Modeling errors

OP	$\Omega Q^{\gamma+1}$	$\theta_1 Q^2$	$\theta_{11}Q^2+\theta_{21}Q$	$\theta_2 Q^{1.852}$
A	**2.4518e-06**	-0.3588	-0.0541	0.0770
B	**-0.0032**	-0.1789	-0.0549	0.0191
C	0.0649	0.0419	**0.0057**	0.0420
D	**0.0207**	0.1086	-0.0538	-0.0345
E	**0.0188**	0.1820	-0.0768	-0.0577
F	**0.0127**	0.2175	-0.1064	-0.0744
G	**0.0114**	0.2275	-0.1307	-0.0765
H	**-0.0078**	0.1922	-0.1681	-0.0874
I	**-0.0283**	0.1332	-0.1962	-0.0924
J	-0.0529	**0.0527**	-0.2097	-0.0952

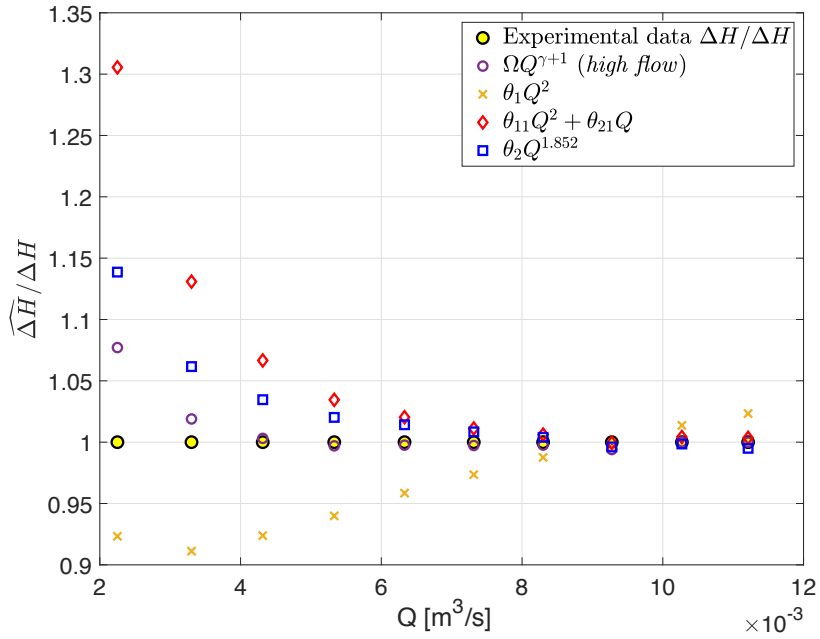


Fig. 8.7: Comparison of different head-loss models in steady state

The second evaluation is a comparison between the estimation results provided by the proposed approach and the results obtained with the next estimation methodology. The MATLAB instruction `fit` was used to find the parameters of the power law by interpolating the operation points B,C and D, which correspond to the same operation range of the oscillatory flow used for the estimation with the state observer: the *high flow*. Check Fig. 8.4 to verify this fact. The results were $\Omega_{fit} = 68730$ and $\gamma_{fit} = 0.8717$.

The relative errors of the fitting for each operation point by using both methods are displayed in Table 8.5. Note that the adjustment with the power law estimated with the proposed approach has small errors

Table 8.5: Approach errors

OP	Online-based approach	MATLAB(B:D)
A	$\mathbf{2.451}\times10^{-6}$	0.0108
B	**-0.0032**	-0.0146
C	0.0649	**0.0358**
D	0.0207	**-0.0204**
E	**0.0188**	-0.0293
F	**0.0127**	-0.0376
G	**0.0114**	-0.0368
H	**-0.0078**	-0.0498
I	**-0.0283**	-0.0612
J	**-0.0529**	-0.0741

in almost all points of operation, except points C and D. This fact is a clear evidence that the proposed method estimates accurately the values for the power law.

8.6 Relevance of head loss calibration in leak location

One of the requirements for implementing a model-based fault diagnosis system is to have a suitable and well-calibrated model. Furthermore, when we speak about calibration in the context of modeling and analysis of dynamic systems, we refer to the identification of systems, which is the process by which the parameters of a model are adjusted, using data measured from the real process. Thus, the error between the model and the process is the smallest possible using some metric.

In the context of *Fault Detection and Identification*, there are parameters or quantities that can most notably affect the diagnosis of a fault, so their estimation during calibration should be emphasized to obtain the best accurate diagnosis. In the case of leak detection and location, this quantity is head loss.

To verify the above remark, numerical simulations of a leak location algorithm based on a state observer were performed. The state observer is based on the model, which describes the flow in two different sections of the pipeline: the section to the left of the leak (namely $\Delta z = z_\ell - z_0$) where the head loss is $\Delta H = H_{in} - H_\ell$ and the section to the right of the leak (namely $\Delta z = L - z_\ell$) where the head loss is $\Delta H = H_\ell - H_{out}$:

$$\dot{Q}_{in} = \frac{\theta}{z_\ell - z_0}(H_{in} - H_\ell) - \alpha Q_{in}^{\gamma+1}, \tag{8.18}$$

$$\dot{Q}_{out} = \frac{\theta}{L - z_\ell}(H_\ell - H_{out}) - \alpha Q_{out}^{\gamma+1}. \tag{8.19}$$

where Q_{in} and Q_{out} are the inlet an outlet flow rates, respectively. H_{in} and H_{out} are the upstream and downstream hydraulic heads, respectively. $\theta = gA_r$, α is a factor involving the friction losses, γ is the head loss exponent, $z_0 = 0$ is the origin coordinate (the upstream end) of the pipeline, z_ℓ is the leak coordinate (position) and H_ℓ is the pressure head at the leak junction.

If H_ℓ is replaced by any of the following equations obtained from (8.18) and (8.19) in steady state

$$H_\ell = H_{in} - \frac{\alpha}{\theta} Q_{in}^{\gamma+1} z_\ell, \;\; H_\ell = \frac{\alpha}{\theta} Q_{out}^{\gamma+1} (L - z_\ell) + H_{out}, \tag{8.20}$$

and by defining the following new state variables $x_1 = Q_{in} - Q_{out}$ and $x_2 = 1/z_\ell$, then the following second-order system can be obtained from (8.18) and (8.19):

$$\begin{aligned} \dot{x}_1 &= x_2 \left[\theta (H_{in} - H_{out}) - L \left(\alpha Q_{out}^{\gamma+1} \right) \right] - \alpha \left(Q_{in}^{\gamma+1} - Q_{out}^{\gamma+1} \right), \\ \dot{x}_2 &= 0. \end{aligned} \tag{8.21}$$

Notice that (8.21) has the following general form:

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & u \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \varphi \\ 0 \end{bmatrix}, \\ y &= x_1, \end{aligned} \tag{8.22}$$

where $u = \left[\theta (H_{in} - H_{out}) - L \left(\alpha Q_{out}^2 \right) \right]$ and $\varphi = -\alpha \left(Q_{in}^2 - Q_{out}^2 \right)$ are known smooth functions.

For system (8.22), the following state observer for locating single leaks was proposed in Torres et al. (2019):

$$\begin{aligned} \begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & u \\ 0 & 0 \end{bmatrix}}_{A} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} + \begin{bmatrix} \varphi \\ 0 \end{bmatrix} + \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} e, \\ \hat{y} &= \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \hat{x}, \end{aligned} \tag{8.23}$$

where $\hat{x}_1 = \hat{Q}_{in} - \hat{Q}_{out}$ and $\hat{x}_2 = 1/\hat{z}_\ell$ are the states to be estimated. Moreover, $e = y - \hat{x}_1$ is the observation error. K_1 and K_2 are elements of the gain vector K, which can be calculated as follows: $K = S^{-1} C^T$, where S is the solution of the following matrix Lyapunov differential equation:

$$\dot{S} = -\lambda S - A^T S - SA + C^T C, \tag{8.24}$$

with $\lambda > 0$ and $S(0) > 0$.

To recover the estimation of the leak position from the state observer, $\hat{x}_2$ must be inverted: $\hat{z}_\ell = 1/\hat{x}_2$.

The fundamental objective of the numerical simulations was to investigate the result of the leak location by taking into account uncertainty in the parameter α, which is a key quantity used for describing the head loss in a duct.

The uncertainty was modeled by a uniform probability density function (p.d.f.) given by $\mathscr{U}(a,b)$, where $a \sim \alpha - 0.2\alpha$ and $b \sim \alpha + 0.2\alpha$. A histogram of the p.d.f. for α is given in Fig. 8.8.

The leaks were simulated at four different positions: $z_\ell = [122.79 \quad 81.86 \quad 40.93 \quad 20.465]$ (m), where $L = 163.72$ (m) is the total length of the simulated pipeline. The rest of the physical parameters used to simulate the pipeline are given in Table 8.1.

The results of the leak location with friction uncertainty for the different position cases are summarized in Table 8.6, which shows the estimate clearly deviates when the value of α is not exactly known. On

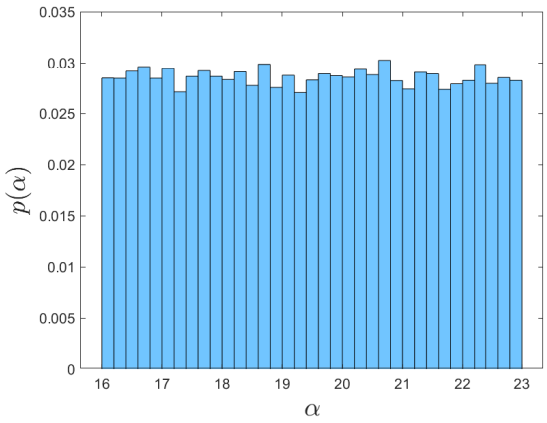


Fig. 8.8: α uncertainty distribution

the contrary, in Fig. 8.9 is shown that the true leak position's values are estimated when there is no uncertainty in friction.

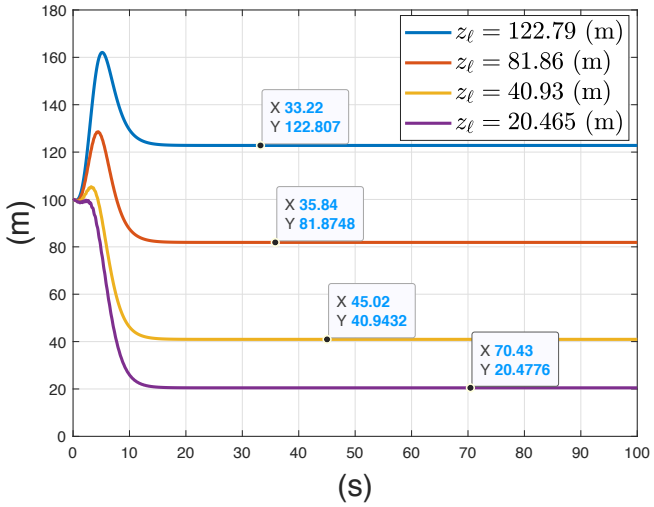


Fig. 8.9: Leak position estimation with the true value for μ

Table 8.6: Mean and standard deviation of the leak position estimations

Leak position (m)	**Mean** (m)	**Standard Deviation** (m)
20.465	20.36	8.52
40.93	40.41	7.20
81.86	80.70	5.30
122.79	120.917	5.07

8.7 Conclusions

This chapter has proposed a method based on the rigid water column model to calibrate the parameters of a power law that describes the head loss of a pipeline. The method can be applied using real-time information from, for example, SCADA or IoT systems. Some advantages of the proposed method are listed below.

- The method permits calibrating a head loss model with only two parameters to be identified, instead of calibrating a model with more than two parameters, such as the DW equation or the HW formula.
- The method is an alternative to calibrate a head loss model even though the physical parameters of the pipeline, such as roughness or the cross-sectional area, are unknown.
- It allows finding a good set of parameters so that the power-law model can be adjusted accurately to the head loss curve. Regarding this advantage, a comparison between the parameter estimation was presented with the new proposition and the offline least squares algorithm performed using MATLAB.
- It provides similar results regardless of the pipeline operation region in which the oscillatory flow is generated for the calibration.
- The method provides a good set of parameters so that the power-law model accurately describes the head loss at all the operating points, regardless of whether the oscillatory flow has been generated around an operating point in a small region. This is an advantage over other methods that require information (pressure and flow rate) from all the operation points to obtain a satisfactory set of parameters.
- It can be applied to a long pipeline by measuring pressure at two different pipeline coordinates delimiting a short section of the pipeline.
- It can be used online together with leak diagnosis algorithms.

References

Adamkowski, A. and Lewandowski, M. (2006). Experimental examination of unsteady friction models for transient pipe flow simulation. *Journal of Fluids Engineering*, 128(6):1351–1363.

Bergant, A., Ross Simpson, A., and Vìtkovsk, J. (2001). Developments in unsteady pipe flow friction modelling. *Journal of Hydraulic Research*, 39(3):249–257.

Besançon, G. (2007). *Nonlinear Observers and Applications*. Springer.

Besançon, G. (2016). A link between output time derivatives and persistent excitation for nonlinear observers. *IFAC-PapersOnLine*, 49(18):493–498.

Brkić, D. (2011). Review of explicit approximations to the Colebrook relation for flow friction. *Journal of Petroleum Science and Engineering*, 77(1):34–48.

Cabrera, E., Garcia-Serra, J., and Iglesias, P. L. (1995). Modelling water distribution networks:from steady flow to water hammer. In *Improving effic. & reliability in water distribution syst.*, pages 3–32. Springer.

Chaudhry, M. H. (1979). *Applied Hydraulic Transients*. New York: Van Nostrand Reinhold.

Datta, R. and Sridharan, K. (1994). Parameter estimation in water-distribution systems by least squares. *Journal of Water Resources Planning and Management*, 120(4):405–422.

Díaz, S., Mínguez, R., and González, J. (2017). Calibration via multi-period state estimation in water distribution systems. *Water resources management*, 31(15):4801–4819.

Farshad, F. F., Pesacreta, T. C., Garber, J. D., and Bikki, S. (2001). A comparison of surface roughness of pipes as measured by two profilometers and atomic force microscopy. *Scanning*, 23(4):241–248.

Gao, T. (2017). Roughness and demand estimation in water distribution networks using head loss adjustment. *Journal of Water Resources Planning and Management*, 143(12):04017070.

Hitzer, E. (2001). Early works on the Hagen-Poiseuille flow. *Memoirs-faculty of Engineering Fukui University*, 49(1):45–54.

Islam, M. R. and Chaudhry, M. H. (1998). Modeling of constituent transport in unsteady flows in pipe networks. *Journal of Hydraulic Engineering*, 124(11):1115–1124.

Kumar, S. M., Narasimhan, S., and Bhallamudi, S. M. (2010). Parameter estimation in water distribution networks. *Water resources management*, 24(6):1251–1272.

Larock, B. E., Jeppson, R. W., and Watters, G. Z. (1999). *Hydraulics of pipeline systems*. CRC press.

Liou, C. P. (1998). Limitations and proper use of the Hazen-Williams equation. *Journal of Hydraulic Engineering*, 124(9):951–954.

Moore, W. (1959) Relationships between pipe resistance formulas. *Journal of the Hydraulics Division*. 85(3): 25–41

Nault, J. and Karney, B. (2016). Improved rigid water column formulation for simulating slow transients and controlled operations. *Journal of Hydraulic Engineering*, 142(9):04016025.

Ostfeld, A., Salomons, E., Ormsbee, L., Uber, J. G., Bros, C. M., Kalungi, P., Burd, R., Zazula-Coetzee, B., Belrain, T., Kang, D., et al. (2011). Battle of the water calibration networks. *Journal of Water Resources Planning and Management*, 138(5):523–532.

Puig, V., Ocampo-Martínez, C., Pérez, R., Cembrano, G., Quevedo, J., and Escobet, T. (2017). *Real-time Monitoring and Operational Control of Drinking-Water Systems*. Springer.

Reddy, P. N., Sridharan, K., and Rao, P. (1996). WLS method for parameter estimation in water distribution networks. *Journal of Water Resources Planning and Management*, 122(3):157–164.

Rojas, J., Verde, C., and Torres, L. (2018). On-line head loss identification for monitoring of pipelines. In *10th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, Warsaw, Poland.

Savic, D. A., Kapelan, Z. S., and Jonkergouw, P. M. (2009). Quo vadis water distribution model calibration? *Urban Water Journal*, 6(1):3–22.

Scola, I. R., Besançon, G., and Georges, D. (2018). Optimizing Kalman optimal observer for state affine systems by input selection. *Automatica*, 93:224–230.

Sletfjerding, E. and Gudmundsson, J. S. (2003). Friction factor directly from roughness measurements. *Journal of energy resources technology*, 125(2):126–130.

Torres, L., Besancon, G., and Georges, D. (2012). EKF-like observer with stability for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 57(6):1570–1574.

Torres, L. and Verde, C. (2018). Nonlinear estimation of a power law for the friction in a pipeline. *IFAC-PapersOnLine*, 51(13):67–72.

Torres, L., Verde, C., and Rojas, J. (2019). Minimal-order observers for locating leaks in a pipeline with a branch. *IFAC-PapersOnLine*, 52(23):67–72.

Verde, C. and Torres, L. (2017). *Modeling and Monitoring of Pipelines and Networks: Advanced Tools for Automatic Monitoring and Supervision of Pipelines*, volume 7. Springer.

White, F. M. (1999). Fluid mechanics, wcb. *Ed McGraw-Hill Boston*.

Xu, X. and Karney, B. (2017). An overview of transient fault detection techniques. *Modeling and monitoring of pipelines and networks*, pages 13–37.

1 Glossary

According to *SAFEPROCESS* convention of the International Federation of Automatic Control, here the main notation and terminology used in the monograph are described. This convention is taken from Blanke et al. (2006) for fault diagnosis and control. The glossary for the degradation analysis is taken from Breton and Daigle (2019). One can also consult the references Puig et al. (2004a), Puig et al. (2004b) and Verde et al. (2013) for the notation in Spanish.

Fault Detection

- **Abrupt fault**: Fault with effects that appear suddenly. An example is a fault modeled by a discontinuous function.
- **Discrepancy:** An abnormal behavior of a physical value or inconsistency that alter physical values and the relationship between them.
- **Failure:** Permanent interruption of a system's ability to perform a required function under specified operating conditions.
- **Fault:** Unpermitted deviation of at least one characteristic property or one system parameter from its standard condition.
- **Fault detection:** Determination of faults present in a system and time of detection.
- **Fault diagnosis:** Determination of kind, size, location, and time of occurrence of a fault. Fault diagnosis includes fault detection, isolation and estimation.
- **Fault effect analysis:** Study of the fault consequence and its is propagation throughout the system.
- **Fault estimation:** Calculation of the size and behavior of a fault over the time.
- **Fault isolation:** Determination of kind, location and time occurrence of a fault. It follows fault detection.
- **Fault mode:** Particular way in which a fault can occur. Each fault usually has associated specific effects.
- **Fault modeling**: A quantitative or qualitative mathematical model that describes the fault behavior.
- **Incipient fault:** A fault with effects that occur in a progressive way. As example, a fault modeled by a time function started with very small deviation from 0 as a ramp.

Fault Diagnosis

- **Analytical redundancy**: The use of two o more, but not necessary identical, ways to determine a variable, where one way uses a mathematical process model in analytical form.

- **Redundancy:** The existence of more than one set of components or data to know the status and characteristics of a system.
- **Hardware redundancy:** The use of two o more independent devices or instruments to estimate a specific function.
- **Residual**: Signals carrying fault information based on a deviation between measurement and model-based computations. A residual describes a degree of consistency between the real and modeled system.
- **Residual evaluation:** Analysis of the residue in order to detect, isolate or identify faults.
- **Residual generation:** algorithm to determine a residual from the known measurements and system's model.
- **Threshold:** Limit value of a residual deviation, so if it is exceeded, a fault is declared. It denotes the residual value from which the existence of the fault is considered.
- **Vote:** Decision-making process based on joint indicators for a fault diagnosis.

Supervision

- **Monitoring:** It is the function of a real-time continuous record system, which can analyze a process's behavior to indicate anomalies. Another term used for this concept is **surveillance** and state observation.
- **Plant-wide supervision**: It is the activity that ensures that the trajectories of the important variables of a process follow the reference trajectories in an adequate way. This terminology is frequently used for a global supervision.
- **Supervision:** Monitoring of a physical system and taking appropriate actions to maintain the operation in the case of faults.
- **Supervisor:** Human or artificial operator who performs the supervision tasks by using fault diagnostic systems, determines the fault and performs corrective actions in the presence of faults.
- **Supervisory:** A function that performs supervision by results of fault diagnosis determines remedial action when needed and executes corrective actions to handle faults. This definition is also used for local supervision.

Fault-Tolerant Control

- **Active fault-tolerance control:** A control system where the faults are explicitly detected and accommodated. This feature is opposite of a passive fault-tolerant system.
- **Admissible control laws**: Set of algorithms that can be implemented to solve a certain control problem in fault condition.
- **Control objective:** Formulation of a certain control task by using requirements and assumptions for the system variables.
- **Control constraints**: Set of functional relations that must satisfy a device or system imposed by physical law or natural laws. It permits the variables to take certain values in the variable space.
- **Fault accommodation:** A change in the system's behavior such that a fault does not produce an undesired failure, and it is implemented in the two forms below:
  - A corrective action based on a system operation change avoiding that a certain failure leads to an undesired final effect.

  - An adjustment on the parameters of the control law to avoid the fault effects. In general the input and output of the controller remains the same, and the control objective is achieved even with a degraded performance.

- **Fault-tolerance system**: A systems where a fault is accommodated with or without performance degradation, and a singles fault does not develop into a subsystem or system failure.
- **Passive fault-tolerance:** A system where faults are not explicitly detected and accommodated, but the controller is designed to be insensitive to a certain set of faults.
- **Reconfiguration:** A change in the input and output of the controller by changes in the structure or its parameters, but the control performance could be degraded.
- **Standard control problem:** It consists of the selection of a control law within an admissible set such that the system satisfies the control objective, and, at the same time the physical constraints are not violated.

Failure Prognosis

- **Monotonicity:** A metric that quantifies the monotonic trend in a prognostic feature as the system evolves toward the failure state.
- **Prognostics:** Estimation of the current system state, and future load and uncertainty. This task predicts at what time a component will reach an specific state.
- **Prognosability:** A metric of the variability of a feature at failure based on experiments.
- **Remaining Useful Life:** Time from the current instant to the time of failure or system health loss. Some authors consider the remaining time to replace or repair a component.
- **Trendability:** A metric of the similarity between trajectories of a feature by several run-to failure experiments.

This page intentionally left blank

References

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2006). *Diagnosis and Fault Tolerant Control*. 2nd. Ed. Springer, Berlin.

Breton, A. and Daigle, M. (2019). *Fault Diagnosis of Dynamic Systems*, chapter Fundamentals of Prognostics, pages 409–432. Springer, Berlin.

Puig, V., Quevedo, J., Escobet, T., Morcego, B., and Ocampo, C. (2004a). Control tolerante a fallos (parte i): Fundamentos y diagnóstico de fallos. *Revista Iberoamericana de Automática e Informática Industrial*, 1(1):15–31.

Puig, V., Quevedo, J., Escobet, T., Morcego, B., and Ocampo, C. (2004b). Control tolerante a fallos (parte ii): Mecanismos de tolerancia y sistema supervisor. *Revista Iberoamericana de Automática e Informática Industrial*, 1(2):5–25.

Verde, C., Gentil, S., and Morales-Menéndez, R. (2013). *Monitoreo Y Diagnóstico Automático De Fallas En Sistemas Dinámicos*. Trillas.

2 Acronyms

This section describes abbreviations frequently used in the text according to the standard of the *International Federation of Automatic Control, Safe Process*, and Mechanical Engineering communities.

AE	Autoencoder
AI	Artificial Intelligence
AISI	American Iron and Steel Institute
ANN	Artificial Neural Networks
ANOVA	Analysis of Variance
ARR	Analytic redundant relation
BLSTM	Bidirectional Long Short-Term Memory Neural Network
BN	Bayesian Networks
CCD	Central Composite Design
C-MAPSS	Commercial modular aero-propulsion system simulation
CNC	Computer Numerical Control
CPOS	Cutting Parameters Optimization and System
CTWC	Cutting Tool Wear-Condition
CUSUM	Accumulative summation
DNN	Deep neural network
DoE	Design of Experiments
EKF	Extended Kalman filter
EOL	End of life
FDI	Fault Detection and Isolation
GA	Genetic Algorithm
GONNS	Genetically Optimized Neural and Network System
HMM	Hidden Markov Model
HSM	High-Speed Machining
IIoT	Industrial Internet of Things
KF	Kalman Filter
LP	Linear Programming
LSE	Least Squares Estimation

MDP	Markov Decision Process
MFCC	Mel Frequency Cepstrum Coefficients
MIMO	Multiple Inputs Multiple Outputs
ML	Machine Learning
MSE	Mean Square Error
NMPM	Neural Milling Process Model
NSM	Neural Simulator Model
PCA	Principal Component Analysis
RG	Residual Generator
RSM	Response Surface Methodology
RUL	Remaining useful life
SCADA	Supervisory Control and Data Acquisition
SISO	Single-input Single-output
SVDD	Support vector data description

3 Variable and Parameter Notation

Lists of abbreviations used in the text according to the standard of the *SAFEPROCESS* community of the *International Federation of Automatic Control* .

Model and Variables	**Description**
$\dot{x}(t) = f_m(x(t), u(t), \theta, \bar{f}(t), f(t))$ $y(t) = h(x(t), u(t), \theta, \bar{f}(t), f(t))$	Nonlinear state model
$\dot{x}(t) = Ax(t) + Bu(t) + E_1\bar{f}(t) + F_1 f(t) + w(t)$ $y(t) = Cx(t) + Du(t) + E_2\bar{f}(t) + F_2 f(t) + v(t)$	Linear state model with additive disturbance
$\bar{f}(t) \in \Re^d$	Disturbance or fault without interest
$f(t) \in \Re^f$	Fault vector
$\theta \in \Re^s$	Parameter vector
$u(t) \in \Re^m$	Input vector
$x(t) \in \Re^n$	state vector
$y(t) \in \Re^d$	Output vector
a_{ij}	coefficient of matrix A
$G(s)$	Transfer matrix

This page intentionally left blank

This page intentionally left blank

Las Series del Instituto de Ingeniería describen los resultados de algunas de las investigaciones más relevantes de esta institución. Con frecuencia son trabajos in extenso de artículos que se publican en revistas especializadas, memorias de congresos, etc.

Cada número de estas Series se edita con la aprobación técnica del Comité Editorial del Instituto, basada en la evaluación de árbitros competentes en el tema, adscritos a instituciones del país y/o el extranjero.

Actualmente hay tres diferentes Series del Instituto de Ingeniería:

SERIE INVESTIGACIÓN Y DESARROLLO

Incluye trabajos originales sobre investigación y/o desarrollo tecnológico. Es continuación de la Serie Azul u Ordinaria, publicada por el Instituto de Ingeniería desde 1956, la cual actualmente tiene nueva presentación y admite textos en español e inglés.

SERIE DOCENCIA

Está dedicada a temas especializados de cursos universitarios para facilitar a estudiantes y profesores una mejor comprensión de ciertos temas importantes de los programas de estudio.

SERIE MANUALES

Abarca manuales útiles para resolver problemas asociados con la práctica profesional o textos que describen y explican el estado del arte o el estado de la práctica en ciertos temas. Incluye normas, manuales de diseño y de laboratorio, reglamentos, comenta rios a normas y bases de datos.

Las Series del Instituto de Ingeniería pueden consultarse gratuitamente desde la dirección electrónica del Instituto http://www.ii.unam.mx (http://aplicaciones.iingen.unam.mx/ConsultasSPII/Buscarpublicacion.aspx) y pueden grabarse o imprimirse en formato PDF desde cualquier computadora.